# An Information-Extraction System for Urdu—A Resource-Poor Language

SMRUTHI MUKUND and ROHINI SRIHARI State University of New York at Buffalo and ERIK PETERSON Janya, Inc.

There has been an increase in the amount of multilingual text on the Internet due to the proliferation of news sources and blogs. The Urdu language, in particular, has experienced explosive growth on the Web. Text mining for information discovery, which includes tasks such as identifying topics, relationships and events, and sentiment analysis, requires sophisticated natural language processing (NLP). NLP systems begin with modules such as word segmentation, part-of-speech tagging, and morphological analysis and progress to modules such as shallow parsing and named entity tagging. While there have been considerable advances in developing such comprehensive NLP systems for English, the work for Urdu is still in its infancy. The tasks of interest in Urdu NLP includes analyzing data sources such as blogs and comments to news articles to provide insight into social and human behavior. All of this requires a robust NLP system. The objective of this work is to develop an NLP infrastructure for Urdu that is customizable and capable of providing basic analysis on which more advanced information extraction tools can be built. This system assimilates resources from various online sources to facilitate improved named entity tagging and Urdu-to-English transliteration. The annotated data required to train the learning models used here is acquired by standardizing the currently limited resources available for Urdu. Techniques such as bootstrap learning and resource sharing from a syntactically similar language, Hindi, are explored to augment the available annotated Urdu data. Each of the new Urdu text processing modules has been integrated into a general text-mining platform. The evaluations performed demonstrate that the accuracies have either met or exceeded the state of the art.

Categories and Subject Descriptors: H.0 [Information Systems]: General

General Terms: Algorithms, Experimentation, Languages, Performance

Additional Key Words and Phrases: Urdu natural language processing, named entity tagging, part of speech tagging, shallow parsing, transliterations, bootstrap learning, text mining

Author's address: S. Mukund, CEDAR, UB Commons, State University of New York at Buffalo, Amherst, NY, 14280; email: smukund@buffalo.edu.

Permission to make digital or hard copies part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from the Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or permissions@acm.org.

© 2010 ACM 1530-0226/2010/12-ART15 \$10.00 DOI: 10.1145/1838751.1838754. http://doi.acm.org/10.1145/1838751.1838754.

#### **ACM Reference Format:**

Mukund, S., Srihari, R., and Peterson, E. 2010. An information-extraction system for Urdu—A resource-poor language. ACM Trans. Asian Lang. Inform. Process. 9, 4, Article 15 (December 2010), 43 pages. DOI = 10.1145/10.1145/1838751.1838754. http://doi.acm.org/10.1145/1838751.1838754.

## 1. INTRODUCTION

The increase in the amount of multilingual data in the form of electronic text has ignited a slew of efforts toward monolingual and crosslingual information retrieval. Specifically, Indic and related languages have received considerable attention in the past few years. For instance, FIRE [Mitra and Majumder 2008] (Forum for Information Retrieval Evaluation) is a new forum that concentrates on showcasing research in important Indian languages. Though FIRE focuses on languages like Hindi and Bengali, it does not cover Urdu. We argue that FIRE should consider research in Urdu, as this language has a large number of speakers in the world (48,062,000 speakers in India alone as of 1997 [Ethnologue 2002]). Urdu has also gained political prominence due to its close relation to the Islamic world, clearly indicating its importance. It is a fascinating language to study given the ethnic and geographical diversity of its speakers.

Analyzing text written in the native language of a society gives us valuable insight into the cultural, political, and social conditions of that society. To facilitate information discovery, there is a need to work on advanced multilingual information extraction (IE) systems. Tools for word segmentation, part of speech (POS) tagging, named entity (NE) tagging, and shallow parsing provide the basic structural information needed for further analysis. In this work we attempt to develop an end-to-end IE system that provides all the above mentioned tools for Urdu, a less commonly taught language (LCTL) [Janus 1997].

Providing the basic NLP tools for Urdu presents many challenges. Before we can address the challenges, we need to address an important question that often arises while discussing natural language processing for Urdu. If Urdu and Hindi (a language commonly spoken in India) are treated as similar languages [King 1995], can the NLP tools developed for Hindi be applied to Urdu? Hindi and Urdu are two standardized forms of Hindustani, an Indo-Aryan language that contains several closely related dialects of northern India and Pakistan. Urdu is considered to be very similar to Hindi as it shares its phonological, morphological, and syntactic structure with Hindi. Both these languages evolved from Sanskrit and share the common Khari Boli dialect. They are free word order languages and follow a general SOV (Subject-Object-Verb) structure. But despite the similarities, NLP tools developed for Hindi cannot be used as is for Urdu [Russell 1996]. The three main differences that hinder the direct use of Hindi tools for Urdu are the following.

(1) Script difference. Urdu has the *nastaleeq* and *nasq* style of writing that is similar to Arabic which flows from right to left [Ahmad et al. 2007]. Hindi

 $<sup>^1</sup>$ See http://www.ethnologue.com/14/show\_language.asp?code=URD.

- on the other hand, is written in Devanagiri script which flows from left to right.
- (2) Vocabulary difference. The vocabulary of Urdu is highly influenced by Arabic and Persian, while Hindi is influenced by Sanskrit.
- (3) Missing diacritics problem. The problem of missing diacritics in Urdu prevents direct use of Hindi tools that consider token-level features.

For tasks that require word level features, using Hindi tools on Urdu fails to generate accurate results (see Sections 6 and 7). Hindi script is space delimited and hence does not encounter the problem of word segmentation. Since word segmentation is the precursor to any NLP task, it affects the accuracy of subsequent NLP tasks. Another interesting issue is that of missing diacritics. The addition or deletion of diacritics is based on the writer and is not always consistent. To make NLP modules such as POS tagger, NE tagger, etc., more accurate, these modules are trained on data normalized for diacritics. However, since Hindi does not suffer from the missing diacritic problem, Hindi taggers—which consider token level features—when used on Urdu fail to produce impressive results. But, using Hindi resources to generate suitable Urdu taggers that depend on syntactic information (e.g., shallow parsers) holds promise (see Section 7).

Learning methods used in the literature for each of these NLP tasks require annotated data. For English, the Penn Treebank corpus consisting of more than 4.5 million words is used to train a state-of-the-art POS tagger [Marcus et al. 1994]. The Chinese Treebank project [Xue et al. 2005] provides a well segmented, POS tagged corpus of more than 500,000 words for the Chinese language. Arabic is also gaining significant attention in the research community and an Arabic Treebank [Maamouri et al. 2004] consisting of part of speech and morphological annotations is available. The Linguistic Data Consortium has also released the Korean Treebank [Han et al. 2002]. An excellent compilation of resources is available at this Web site.<sup>2</sup> Unfortunately Urdu is not as privileged as these languages when it comes to the availability of annotated resources. Though there exists the Hindi-Urdu Treebank initiative [Bhatt et al. 2009], data is still not available for research. As of today, POS annotated data for Urdu is provided by the Center for Research in Urdu Language Processing (CRULP),<sup>3</sup> IIIT Hyderabad<sup>4</sup> and The EMILLE Project [Hardie 2003]. But these data sets are based on different tagsets. The same is true with the two NE annotated corpora provided by IJCNLP (2008) and Computing Research Laboratory (CRL).<sup>5</sup> Apart from the EMILLE dataset, other datasets are very limited in terms of size. Developing good taggers that use these limited resources is not a trivial task. Furthermore, incorporating all these NLP tools into a single comprehensive system is a challenging task.

<sup>&</sup>lt;sup>2</sup>See http://tiny.cc/corpora.

<sup>&</sup>lt;sup>3</sup>See www.crulp.org.

<sup>&</sup>lt;sup>4</sup>See http://ltrc.iiit.ac.in/showfile.php?filename=downloads/.

<sup>&</sup>lt;sup>5</sup>See crl.nmsu.edu.

## 2. RELATED WORK

Developing better methods for segmenting continuous text into words is important for improving the automatic processing of Asian languages [Wang et al. 2007]. For tasks such as machine translation and information retrieval, the fundamental text unit is a word token. All operations such as morphological analysis, semantic reasoning, etc., are performed on this word token. Similar to Chinese, Japanese, and Korean (CJK languages) [Choi et al. 2009], Urdu too has the problem of word segmentation [Durrani 2007]. Words in Urdu are written continuously without the use of word delimiters (white space, comma, quote, and period). Trying to retrieve word tokens from such text is non-trivial. The approaches used for word segmentation can be categorized as dictionary-based [Wang et al. 2007], machine-learning based [Gao et al. 2005], and transformation-based [Xue 2003]. Approaches that combine all three have also been explored [Sproat and Shih 2002]. In this work, we propose a technique that attempts to mark word segmentation boundaries by combining Hidden Markov Model (HMM) based learning with dictionary lookups (see Section 5).

Obtaining the POS information of word tokens is an important NLP task that offers great benefits. Tasks that use semantic information such as sentiment analysis, agent-target identification, role labeling, etc., consider the significance of the content words [Nicholls and Song 2009] for analysis. Shah and Bhattacharyya [2002], Kanaan et al. [2005], and Karimpour et al. [2008] have conducted studies that evaluate the importance of parts of speech for information retrieval. The POS information associated with a word helps to differentiate word senses [Wilks and Stevenson 1996]. POS taggers have been developed using primarily two techniques: by relying on only linguistic rules in an unsupervised manner [Brill 1992], and by using machine learning algorithms such as Hidden Markov Model (HMM) [Ekbal and Bandyopadhyay 2007], Maximum Entropy (MaxEnt) [Dandapat et al. 2007; Ratnaparkhi 1996] and Conditional Random Field (CRF) [Avinesh and Karthik 2007; Lafferty et al. 2001]. The tagging technique used and the efficiency of the resulting tagger depend on the tagset that is used. The granularity of the tagset chosen, in-turn depends on the task that it assists. We develop an Urdu POS tagger using a MaxEnt model, the details of which are discussed in Section 6.

Tasks such as machine translation [Garrido-Alenda et al. 2004], thematic role extraction [Shamsfard and Mousavi 2008], and semantic role labeling [Punyakanok et al. 2005] are greatly benefitted by not only POS and NE information but also shallow parse information. Shallow parsing divides text into segments that correspond to certain syntactic units. Shallow parsers are commonly used to perform clause chunking and identifying noun phrases. These shallow parsers are also used to bootstrap a complete parser. In the literature, several techniques have been applied to perform shallow parsing. Ramshaw and Marcus [1995] use transformation-based learning. Daelemans et al. [1999] use memory-based learning. In the shared task for CoNLL-2001 [Erik et al. 2001] several techniques that use HMMs, memory-based learning, boosting, etc., were presented. The chunk and clause information in a

sentence are represented by means of tags. Defining these tagsets is language dependent. Our approach to develop a shallow parser for Urdu is based on the resource sharing technique. Section 7 delves into the details of this method for this work.

NE extraction involves identification and classification of terms in a document that belong to predefined categories such as names of persons, organizations, locations, date/time, etc. This NLP task is very challenging as the named entities are infinite in number and often new entities are coined over time. In information extraction systems, accurate detection and classification of named entities is very important as the named entities help to extract knowledge from texts. NE recognition also finds application in question and answering systems [Moldovan et al. 2002] and machine translation [Babych and Hartley 2003]. Different techniques have been used to develop NE recognizers: rule-based recognizers [Farmakiotou et al. 2000], statistical recognizers such as BBN's HMM-based IdentiFinder [Bikel et al. 1999], and New York University's Max-Ent based recognizer, MENE [Borthwick 1999]. Using CRF [McCallum and Li 2003] for NE recognition in Indian languages [Ekbal and Bandyopadhyay 2009] has also shown promising results. Section 9 describes in detail the technique used in this work to develop an NE tagger for Urdu.

We attempt to develop a complete NLP system that performs the basic language analysis for Urdu, including assimilation of available resources for learning and creation of lexicons for lookups. Our focus is to ensure that each of the text-processing modules makes complete use of the available resources and achieves state-of-the-art accuracy for Urdu. We address the challenges at each stage of text processing and provide suitable solutions to circumvent them. Our word segmentation module achieves an accuracy of 92.64% for space omission errors. The POS tagger we developed has an average F-measure of 88.9%. An F-measure of 89.99% is obtained for the shallow parser we developed using resources from Hindi. The NE tagger developed based on ACE [ACE 2005] guidelines achieves an F-measure of 72.6%. As the next step in the NLP pipeline, we attempt transliterations for person names. These transliterations can aid systems for machine translation and question-answering. To this end, we also attempt to develop a morphological analyzer that can further aid tasks such as semantic role labeling (as shown by Pandian and Geetha [2009]), and word sense disambiguation [Mishra et al. 2009]. The modules at different stages in the NLP pipeline are explained in the sections that follow. But before we look at each module and the associated challenges, let us examine the techniques used to assimilate resources on the web for Urdu. Section 3 will also give the reader an idea of the size of the lexicons that we successfully procured for our system.

## 3. RESOURCES

A large amount of text in Urdu is available on the Web which, when organized and registered in a dictionary, can be extremely valuable. Mining the Web for transliterations and English translations of Urdu words and compiling them into a translation dictionary will significantly and positively affect

the quality of cross-lingual NLP applications, such as machine translation and cross-lingual information retrieval. We have compiled several of the resources into well-categorized lexicons. These lexicons are used to assist POS tagging, NE tagging, and transliteration modules in our NLP pipeline.

Wikipedia<sup>6</sup> is a goldmine of information. There is a growing community of researchers that recognize it as a source of exceptional scale and utility. Many Wikipedia entries have descriptions in several languages. The use of Wikipedia in question and answering systems [Ahn et al. 2004; Buscaldi and Rosso 2006; Ko et al. 2007], in summarization [Baidsy et al. 2008], and in parallel corpora generation [Adafre and Rijke 2006] by exploiting its multilingual property has shown promising results. Richman and Schone [2008] utilize the multilingual characteristics of Wikipedia to annotate a large corpus of text with NE tags. Wikipedia resources for Urdu, though not significant, are growing rapidly. As of November 2009, Urdu Wikipedia had more than 11,500 articles making it a potential source for mining named entities. In this work, we have exploited the multilingual property of Wikipedia and mined Urdu Wikipedia<sup>7</sup> to generate NE lexicons with their respective English transliterations.

In Wikipedia, the articles are organized based on categories. Category entries such as Living people, Births, and Deaths can be used as filters to extract person names into lexicons. Location, Geographical Location, City, and Country, etc., can act as filters for location entities. Similar categories exist for entities that refer to organizations. Links are maintained between articles on the same topic written in different languages across the Wikipedia domain. We exploited this link structure that exists between English and Urdu to mine named entities. The titles of articles on the same topic across the two languages are extracted into a lexicon, and the English title is used as the translation of the Urdu title. The categories are organized based on the standards set by ACE for entity types. For example, categories such as "صدر" (siyasatdan) are grouped as PER\_Individual, "ممالك" (mulk) as GPE\_Nation and "نشر واشاعت" as ORG\_Media.

Another online encyclopedia style Web site, NationMaster,<sup>9</sup> provides Urdu translations for all named entities in articles related to Pakistan and Islam. These translation pairs are also mined and added to the lexicons based on the category filters. We have also mined several name databases on the Web, *urduseek.com and apniurdu.com*, which provide the most common Islamic names along with their English transliterations. These lexicons are used to generate a statistical name transliterator as we describe in Section 10.

Table I gives an overview of the number of entities retrieved using this method for the three main NE categories: Location, Organization, and Person. Other categories such as Facilities with subtags FAC\_Airport, FAC\_Building\_Grounds, and FAC\_Path have a total of 48 entries. Our lexicons are also categorized to indicate colors, Islamic months, Islamic weeks, time,

<sup>&</sup>lt;sup>6</sup>See www.wikipedia.org.

<sup>&</sup>lt;sup>7</sup>See http://ur.wikipedia.org.

<sup>&</sup>lt;sup>8</sup>See http://en.wikipedia.org/wiki/Wikipedia:WikiProject\_Categories.

<sup>&</sup>lt;sup>9</sup>See www.nationmaster.com/.

CATEGORIES	COUNT	CATEGORIES	COUNT	CATEGORIES	COUNT
<b>LOCATION</b>		ORGANIZATION		PERSON	
GPE_State_	169	ORG_Commercial	20	Designation	51
or_Province					
GPE_Nation	192	ORG_Educational	35	English_Name	807
GPE_Population_	50	ORG_Media	10	First_Name	665
Center					
LOC_Celestial	102	ORG_Non_	91	Last_Name	609
		Governmental	91	Last_Name	609
LOC_Land_	100	ORG_Religious	14	Middle_Name	261
Region_Natural					
LOC_Water_	78	ORG	32	Combined_Names	2590
Body					
GaCity	125			Prominent_	647
				Figure	
NeLocs	315				

Table I. NE Categories and Associated Number of Lexicon Entries

directions, and common greetings. We have also maintained lexicons with names of famous personalities such as chief justices, prime ministers, presidents, army generals, and politicians of Pakistan (close to 200 entries).

The lexical categories mentioned above aid in NE recognition (see Section 9). Translations for common Urdu words along with their most probable POS information are also available across different sources on the Web. Waseem Siddiqi<sup>10</sup> (2008) compiled the first offline Urdu-English/English-Urdu dictionary with more than 24,000 entries where each entry is assigned the most probable POS tag with which it is associated. Lexicons are also provided by Computing Research Laboratory (CRL) as a part of their machine translation corpus for Urdu. Lexicons for parts of speech such as adjectives, nouns, and verbs are obtained from the Urdu POS corpus provided by EMILLE. These resources are compiled to one single format and grouped into probable POS categories such as Ga\_JJ, Ga\_NN, and Ga\_VB for adjectives, nouns, and verbs, respectively. Each category has approximately 3,000 words. These lexicons aid the POS tagging model that we describe in Section 6.

# 3.1 Semantex<sup>TM</sup> Architecture

Each of the text processing modules that we present in the NLP pipeline are integrated into Semantex<sup>TM</sup>, a text processing tool provided by Janya Inc.<sup>11</sup> Semantex<sup>TM</sup> is a customizable platform for scalable information extraction. It reflects a hybrid paradigm, supporting both grammar-based and machine-learning components in a pipelined architecture. Documents are converted to an internal tree representation known as a *tokenlist*; subsequent modules (lexical look-up, grammars, or statistical components) modify the tokenlist by adding new features to tokens, creating groups of tokens or links between tokens. The underlying grammar formalism is a tree-walking automaton; this

 $<sup>^{10}\</sup>mathrm{See}\ \mathrm{http://biphost.spray.se/tracker/dict/.}$ 

<sup>&</sup>lt;sup>11</sup>See http://www.janyainc.com/.

has evolved from an earlier finite state transducer model. Grammars are typically regular expressions over token lists, but can access any part of the parse tree through special operators. The machine learning framework supports models such as MaxEnt and CRF. The pipeline can be configured for different extraction tasks, data sources, or languages. Language porting to Urdu has, for the most part, consisted of adding new data resources: lexicons, grammars, and statistical models based on machine learning. An IDE for development, access to the complete English IE system, integration with Lucene/SOLR, and an academic licensing program were additional reasons for it being chosen as the integration platform.

Now that we have access to a fairly large repository of well-categorized lexicons as well as Semantex<sup>TM</sup> framework for seamless integration, we proceed to explain each of the NLP modules that go into the pipeline framework.

#### 4. PREPROCESSING MODULE

The first module in the NLP pipeline that comes even before the word segmentation module is the preprocessing module. This module consists of two subtasks: diacritics removal and normalization.

In Urdu, diacritics/Aerab [Hussain and Afzal 2001] are not consistently marked and their usage is left to the writer's discretion. Even with the same writer, the markings are not uniform. The same is the case with machineprinted Urdu text. Though the virtual keyboards provided for Urdu have options to include diacritics like zer (U+0650) and zabar (U+064E), these options are not commonly used while typing. So a word such as "قابل (qabil  $\sim having$ potential) which contains a zer above "ب" to indicate the vowel "i", is often written as "قابل", without the zer diacritic. But several users exploit the fact that the virtual keyboards have the option of entering the diacritics. It is essential to maintain consistency throughout the text. One approach to attain consistency is to restore the diacritics. The process of restoring diacritics is a very complicated task. Lexicons, annotated corpora used for training, and other resources contain words whose diacritics have to be restored as well. This adds to the complexity of the task. Another method is to consider complete removal of all diacritics. We chose the latter method and applied it to lexical data, training data, as well as incoming data and removed all diacritics (zer, zabar, and pesh) present.

The next filtering task is normalization. This has to be done to keep the Unicode of the characters consistent. Several characters in Urdu have different orthographic forms and these variations cause discrepancies in NLP. UTF-8 standard has multiple representations of several characters to accommodate cross-lingual variants. However, most writers tend to use these variants as they please without a consistent representation. For example, character  $\mathfrak{f}$  has two representations, Unicode value U+0622 and also Unicode values U+0627 + U+0653. There are several other ambiguous cases such as the use of  $\mathfrak{s}$  vs.  $\mathfrak{s}$  (Yeh vs. Alif Maqsura) and the use of  $\mathfrak{s}$  vs.  $\mathfrak{s}$  (Heh vs. Taa Marbuta). It is necessary to convert different representations to one standard form.

Normalization is the process of converting multiple equivalent representations of data to a consistent underlying normal form. CRULP has developed their own normalization tool [CRULP 2007] that provides three different options for normalization: Normalization form D (NFD), Normalization form C (NFC), and Normalization form KD (NFKD). These three options are based on the two equivalences defined by the Unicode Normalization standard: canonical equivalence and compatibility equivalence. The details can be found on the CRULP Web site. 12 In this work, we use a variation of the Normalization form C (NFC) where a decomposed representation of a character is replaced with its composed representation. When applied on a set of 100 Urdu articles downloaded from BBC Urdu<sup>13</sup> and Jang daily<sup>14</sup> (about 12,000 words), we found nearly 90 instances where character representations had to be normalized. We also noticed that a majority of these inconsistencies were in words that were proper nouns. This indicates that though the percentage of inconsistencies is small, normalization has to be performed as these errors can propagate and affect NLP tasks that come later in the pipeline (POS, NE, etc.).

# 5. WORD SEGMENTATION

Word tokens are the fundamental blocks on which NLP tools such as POS tagger, NE tagger, etc., are developed. Generating these word tokens in some languages essentially maps to simply using space and special characters (;, () etc.) as word delimiters (e.g., English, Hindi). However, there are other languages in which determining word delimiters to generate tokens goes beyond determining just the space and the special characters (Thai, Chinese, Urdu, etc.). Urdu is one such language and segmenting a sentence to generate word tokens in this language is not trivial. Urdu suffers from both space insertion and space deletion problems. These issues are solved by the reader who, while reading, groups the characters into words based on contextual knowledge. However, providing such contextual information for a statistical learning algorithm is an extremely difficult and expensive task.

Several different techniques have been proposed for other languages that face similar segmentation challenges. Traditional techniques such as longest and maximum matching strings depend on the availability of a lexicon that holds all morphological forms of a word. Such lexicons are not readily available for Urdu. Statistical techniques apply probabilistic models (usually bigram and trigram models [Papageorgiou 1994]) toward solving the word segmentation problem. Feature-based techniques [Meknavin et al. 1997] that use POS information for tokenization, consider the context around a word for specific words and collocations. There are other models that generate segments by considering word level and syllable level collocation [Aroonmanakul 2002] when developing a learning model for word segmentation. Statistical models that consider character-based, syllable-based, and word-based probabilities have also been

 $<sup>^{12}</sup> See\ http://www.crulp.org/software/langproc/urdunormalization.htm.$ 

 $<sup>^{13}\</sup>mathrm{See}$  http://www.bbc.co.uk/urdu/.

<sup>&</sup>lt;sup>14</sup>See http://www.jang.com.pk/.

shown to perform reasonably well. The word segmentation problem for Thai was solved by Pornprasertkul [1994] using the character-based approach.

However, for a language like Urdu, a model that is purely statistical may fail to yield good results as modeling all erroneous cases is a challenge for Urdu. Space insertion problems are found generally around abbreviations, affixes, suffixes, and proper nouns and space deletion problems are found generally around verbs, proper nouns, auxiliary verbs, and pronouns, making the segmentation issue highly dependent on the morphological and lexical features. A mixed model that considers morphological as well as semantic features of the language facilitates better performance as shown by Durrani and Hussain [2010]. They proposed a word segmentation model that uses a lexicon for proper nouns and a statistical model that trains over the *n*-gram probability of morphemes. Maximum matching string technique is then used to generate word boundaries of the orthographic words that are formed. The generated segments are ranked and the best one is accepted. Space insertion and space deletion problems are separately dealt with. Durrani and Hussain's [2010] solution for space insertion issue associated with compound words and affixes is highly dependent on lexicons. This approach has a combined error detection rate of 85.8%.

We propose a word segmentation model that combines a character-based statistical approach [Pornprasertkul 1994] and grammar rules with lexicon lookups to generate word boundaries. A character in a word can occur in three positions: initial, medial, and final. In Urdu, there are some characters that occur only in certain positions (e.g., " $\cup$ "  $\sim$  noon gunna occurs only at the end of words). The bigrams of each of these characters, based on their positions, are obtained by training over a properly segmented training set. For unknown characters, unknown character models for all the three positions of occurrence are also trained. The following probabilities are estimated and maximized at character level using the Viterbi algorithm.

- (1) Probability of character k being in medial position given character k-1 is in initial position.
- (2) Probability of character k being in final position given character k-1 is in initial position.
- (3) Probability of character k being in final position given character k-1 is in medial position.
- (4) Probability of character k being in medial position given character k-1 is in medial position.
- (5) Probability of character k being in initial position given character k-1 is in final position.

Suitable word boundaries (space delimited) are generated using a combination of morphological rules, lexicon lookups, bigram word probabilities, and bigram HMM character model. Each word thus formed successfully is verified for morphological correctness (rule based) using a sliding window approach. If the word is not valid morphologically, then the window is moved back over three characters and at every step the validity of occurrence of the word is noted.

Similarly, the window is moved three characters ahead and the validity of the word is verified. All words formed successfully are taken and further processed using a language model that considers the bigram occurrence for each word. The unknown word probability is also accounted for. The word with maximum probability is taken as valid in the given context. We also note the number of times transitions occur from syllable sets with consonants to syllable sets with vowels in a word. This number of transitions in a valid word cannot be more than four (decided after observing the errors in the corpus). The Viterbi algorithm used for decoding also accounts for an already existing space.

Some of the rules considered while deciding the word boundaries are given here. A word boundary is formed when:

- (1) the word ends with "o" nun gunna,
- (2) the character transitions over to digits,
- (3) punctuations marks are encountered (- is also included),
- (4) if current character is "alif" and the previous character is "ee" *bari ye*, then the word boundary occurs after "*alif*",
- (5) the word phonemes repeat (reduplication within the orthographic words).

There are other rules that we consider such as

- (1) no two "ye" choti ye come back to back; and
- (2) no characters occur in detached form unless they are initials or abbreviations followed by a period.

In order to train the character transition model and the language model, we use the POS-tagged data set released by CRULP (see Section 6). This data set is well segmented and has a wide coverage of pronouns, verbs, and auxiliaries. We tested our technique on a fairly small data set of 50 sentences (~ 1,700 words) obtained from BBC Urdu. Most errors in our test set are associated with the space omission problem. Many of the auxiliary verbs are combined with the verbs. For example, "- کی گئی ہے "  $\sim ki \; gayi$ hai (has been done) is written as "- کیگئی ہے " ~ has been done. Many of the pronouns are combined with case markers. For examples, " اس کے" ~ us~ki~(his) is written as "سکے"  $\sim his$ . The space insertion problem is predominantly around the abbreviations [Durrani and Hussain 2010]. For example, "بوایس"  $\sim US$  is written as "ایسیو"  $\sim US$  (U.S. is a country name and should be treated as one word). In order to handle these errors, we maintain POS lexicons for most common auxiliary verbs, pronouns, case markers, and determiners. We then apply our algorithm that generates suitable word boundaries using a combination of morphological rules, lexicon lookups, bigram word probabilities, and bigram HMM character model.

We report our analysis only for the space omission problem. Our test set has 136 instances of space omission problems and our algorithm was able to successfully segment 126 instances, thus achieving 92.64% accuracy. This accuracy is close to Durrani and Hussain's [2010] method (94.5%) of using bigrams for space omission. Their best approach generates 96.83%. Unfortunately we

SPECIAL CHARACTERS	HEX	SPECIAL CHARACTERS	HEX
URDU_STOP	06D4	URDU_LEFTPARENTHESIS	0028
URDU_COLON	003A	URDU_RIGHTPARENTHESIS	0029
URDU_DECIMALSEPERATOR	066B	URDU_LEFTBRACKET	005B
URDU_PERCENTAGE	066A	URDU_RIGHTBRACKET	005D
URDU_THOUSANDSEPERATOR	066C	URDU_LEFTCURLY	007B
URDU_COMMA	060C	URDU_RIGHTCURLY	007D
URDU_QUESTION	061F	URDU_DECIMALSEP	066B
URDU_SEMICOLON	061B	URDU_HYPHENMINUS	002D
ARABIC_TATWEEL	0640	URDU_LOWLINE	005F
URDU_QUOTE	2018	URDULESS	003C
URDU_QUOTE1	0027	URDU_GREATER	003E
URDU_EXCLAMATION	0021	URDU_STRONG	0670
URDU_QUESTIONMARK	0022	URDU_SYM1	0604
URDU_DASH	94DB	All Punctuations in ASCII	-

could not run our method on the same test bed as that of Durrani's due to the unavailability of datasets. However, we believe that this difference in the accuracies is because of the following reasons. 1. There are 18 proper nouns in the test set that have the problem of space omission. For example, "general Musharraf" is written as one word. Our lexicons do not account for affixes and proper nouns, and 2. In Urdu, some words can be written in more than one way. This mostly depends on the diacritics and ambiguity between bari and choti "ye". These variations affect the word segmentation.

Efforts are being made by newswire agencies like BBC Urdu and Jang Daily to maintain a format in their rendering of Urdu and to eliminate many issues arising due to improper word segmentation. The Urdu news wire data from BBC clearly is fairly clean in terms of word segmentation and the few errors that remain can easily be removed by simple heuristic rules. But there is the issue of special characters (shown in Table II) attached to the word body (e.g., "(شكاكر)" has to be "(شكاكر)"  $\sim$  "(Chicago)", with a space between the word and the parenthesis) that has to be considered. Hence we use a simple tokenization model that delimits the words based on spaces and the characters mentioned in Table II. This forms the first stage in the Urdu NLP pipeline.

# 6. PART OF SPEECH TAGGER

The accuracy of a statistical model developed for POS tagging not only depends on the domain of the data set used to train the model but also on the tagset used for annotation. We consider three main Urdu tagsets in our discussion: U1 tagset, Sajjad tagset, and CRULP tagset.

The U1-tagset [Leech and Wilson 1999] consists of 282 morpho-syntactic tags. The tags are differentiated based on the tense, gender, and number information. However, the EMILLE corpus is the only corpus that uses this tagset and the size of this corpus is insufficient for simple statistical methods to learn the morphological dependencies for the large number of tags present in this tagset. Hence, we disregard the U1-tagset in our analysis.

The tagset designed by Sajjad [2007] (henceforth referred to as the Sajjad tagset) contains 42 tags and is designed based on the work done by grammarians Schmidt [1999], Haq [1987], Javed [1981], and Platts [1967]. In this tagset, the pronouns are categorized as personal, reflexive, relative, adverbial, and kaf (a pronoun that adds interrogative property in a sentence). Nouns are categorized as common nouns and proper nouns. There also exists a special tag for negative words (e.g., no, not, non). However, the Sajjad tagset has only one tag for verbs, except for auxiliaries that show aspect (AA) and tense (TA) information about the verb (for more details please refer [Sajjad and Schmid 2009]). CRULP [Hussain 2008; Sajjad 2007] has annotated a 110,000-word corpus using this tagset. This corpus is not freely available for download. The Sajjad tagset was discarded due to lack of fine-grained analysis for verbs.

Another tagset (henceforth called the CRULP tagset) developed as a part of the English-Urdu parallel corpus at CRULP shows promise. It follows the Penn Treebank guidelines [Santorini 1990] and has several tags that are assigned to verbs on the basis of their forms and semantics in addition to the tags for auxiliaries for aspect and tense. It also has two shades of proper nouns that help to identify phrase boundaries of compound proper nouns. A WALA tag is assigned to every occurrence (inflected and uninflected) of the word "wala". A wala word often means "the one" and is influenced by the semantics of the phrase that it follows, for example, "the enlightened one". Due to the colloquial usage of the wala word, a new tag is assigned to accommodate its several morpho-syntactic variations. Muaz and Khan [2009] give a more detailed explanation of the wala word and its usage. For details of the CRULP tagset, please refer to the CRULP Web site. 15 We use this tagset in our work for two reasons: (1) the availability of annotated corpus that uses the CRULP tagset, and (2) the existence of the light verb tags VBL and VBLI which is helpful for semantic role labeling [Sinha 2009].

# 6.1 Using Hindi POS Tagger for Urdu

Before we explain the process of generating an Urdu specific POS tagger, we would like to justify our claim that a Hindi specific POS tagger will not yield good results when applied on Urdu.

In order to do this, we first train a Hindi POS tagger using a CRF learning approach. CRF++ tool<sup>16</sup> is used to facilitate this training. Hindi POS annotated data set is obtained from IIIT Hyderabad and is tagged using the tagset released as part of the AnnCorra standards [Bharati et al. 2006]. This

 $<sup>^{15}</sup> See \ http://www.crulp.org/Downloads/ling_resources/parallelcorpus/Urdu%20POS%20Tagset.pdf. <math display="inline">^{16} See \ http://crfpp.sourceforge.net/.$ 

Table III. Results of Hindi Tagger on Urdu Data

Tag	POS	ACT	CORR	MISS	SPUR	PREC	REC	F
DEM	88	122	21	67	101	0.172	0.239	0.2
IJ	1208	648	248	960	400	0.383	0.205	0.267
NN	2744	4144	1932	812	2212	0.466	0.704	0.561
NNP	1093	20	7	1086	13	0.35	0.006	0.013
PRP	426	608	208	218	400	0.342	0.488	0.402
PSP	2482	2241	1878	604	363	0.838	0.757	0.795
QC	551	336	314	237	22	0.935	0.57	0.708
QF	147	19	5	142	14	0.263	0.034	0.06
QO	36	0	0	36	0	0	0	0
RP	41	193	26	15	167	0.135	0.634	0.222
VM	1040	1593	522	518	1071	0.328	0.502	0.397
RB	153	0	0	153	0	0	0	0
VAUX	750	881	346	404	535	0.393	0.461	0.424
VM	1040	1593	522	518	1071	0.328	0.502	0.397

POS: Actual number of tags, ACT: Number of found tags, CORR: Number correctly tagged, eMISS: Number of missing tags, SPUR: Number spuriously tagged, PREC: Precision, REC: Recall, F: F-Measure (%).

data is represented using SSF format [Bharati et al. 2002]. Since representing Urdu data in this format is not trivial, we apply a simple transliteration on the Hindi data set and generate a representation that Urdu data can also be transliterated to. For example, "eka bAra kaswurabA gAMXI bahuwa bImAra ho gaIM" is mapped to "ek bar kasturaba Gandi bahut bimar ho gain". Urdu written in nastaliq form is also changed to a similar representation. For example, "شکاگو کے گٹمین تھیٹر میں" is represented as "shkaago ki gdmyn thytr myn". Clearly this leads to a mismatch in vocabulary as Urdu suffers from the missing diacritics problem, whereas Hindi does not. We applied the double metaphone<sup>17</sup> algorithm on each transliterated word in both data sets and considered the pronunciation information as a feature in our learning model hoping that this in turn would compensate for the missing diacritics problem. (Results without this pronunciation information are bad with an overall Fmeasure of 38.6%). The CRULP tagset was reduced to AnnCorra tagset. All tags from CRULP tagset found suitable tags in the AnnCorra tagset. Most tags were just a one-one mapping and the subcategories for nouns and verbs were collapsed to NN and VM, respectively. The overall F-measure for the Hindi POS tagger on Urdu corpus is 55.5%.

Table III shows the performance of the Hindi POS tagger on Urdu for a few main tags: Nouns (NN), Adjectives (JJ), Verbs (VM), and Adverbs (RB). We notice that several of the nouns are spuriously marked and many are missing. Adverbs are never marked. Most adverbs are marked as nouns (NN) or verbs (VM). The auxiliary verbs are marked as adverbial particles (RP).

Using a Hindi POS tagger on Urdu data fails for the following reasons: (1) when words are considered as features, many of the open class words in Urdu become unknowns for the Hindi tagger; (2) Hindi tagset is not well defined and

<sup>&</sup>lt;sup>17</sup>See http://en.wikipedia.org/wiki/Double\_Metaphone.

cannot account for a lot of morphological inflections that occur in Urdu due to the Persian influence on Urdu; (3) the missing diacritics problem in Urdu affects accurate transliterations, and (4) several nouns and adjectives in Hindi are derived from Sanskrit while this is not true for Urdu. This clearly indicates that the effort involved in developing tagsets and POS taggers specifically for Urdu is not futile.

# 6.2 Urdu-Specific POS Tagger

Muaz and Khan [2009] trained a TnT Tagger [Brants 2000] and a Tree Tagger [Schmid 1994] on two corpora of Urdu tagged with different POS tagsets (Sajjad tagset and CRULP tagset). TnT tagger on Urdu corpus with Sajjad tagset has an accuracy of 93.01% that is comparable to the work done by Sajjad and Schmid [2009]. The accuracy reported on Urdu corpus with CRULP tagset using Tree tagger is 90.49%. In this work, we use two other learning algorithms: CRF and MaxEnt. The annotated corpus<sup>18</sup> contains sentences translated from the Wall Street Journal (WSJ) set of the Penn Treebank corpus. This is tagged with the CRULP tagset and has a total of 3,325 sentences. A tenfold cross-validation test performed on this corpus using the CRF approach generates an average accuracy of 88% with a maximum of 90.57% and a minimum of 81.1%. The MaxEnt approach also produces comparable results with an average accuracy of 88.9%, a maximum of 90.62%, and a minimum of 81.7%. Out of the ten folds, the fold with the minimum accuracy has the maximum number of unknown proper nouns (equal to 62). We report all our findings on this fold as the modifications that we make show significant improvement on this set by boosting its accuracy to 84.1%.

The training set has all sentences belonging to the 00 folder and a few sentences from the 01 folder of WSJ (Penn Treebank) with a total of 2825 sentences. The remaining sentences belong to the test set (about 500 sentences). Since the training time of the MaxEnt model is significantly lower than the training time of the CRF model and the results comparable, MaxEnt model is used to perform further experiments. A simple MaxEnt model which uses the current word, the previous word and the next two words as features, generates an accuracy of 81.7% as shown in Column 2 of Table IV. Most of the errors involve the verb tags (VB, VBL, VBI, VBLI) and the auxiliary tags (AUXT, AUXA). The major confusion with automatic tagging is to differentiate the VBL (light verb) and the VB (verb) tags. Though considered complex predicates [Sinha 2009] VBL words are syntactically very similar to VB words. Several of the cardinals were tagged as nouns owing to syntactic similarity. Many of the missing NNCM (nouns after case markers) were tagged as NN (nouns). Many of these errors are due to the lack of fine-grained features in the learning model. We worked towards improving the efficiency of this tagging model by including lexical features. The gazetteer tags (Ga\_NN, Ga\_VB, Ga\_JJ) from lexicons compiled from different sources (see Section 3) are assigned to the data by performing lookups and these tags are used as features in the learning model.

<sup>&</sup>lt;sup>18</sup>See http://www.crulp.org/software/ling\_resources/UrduNepaliEnglishParallelCorpus.htm.

We also apply several heuristic rules (see Section 6.3) that help in correcting inappropriate tag assignments. The final result obtained has an *F*-measure of 84.1% as shown in Column 3 of Table IV.

#### 6.3 Heuristic Rules for POS Correction

- (1) Our model tags all the Question Words (QW) such as "كيا" kya as pronoun (PR). All such occurrences are assigned QW tag.
- (2) If the word is "كيّ" kya and the previous tag is an adjective (JJ) and the next tag is a phrase marker (PM) then assign a light verb tag (VBL) else assign a verb (VB) tag to the word.
- (4) All valid cardinals were tagged as nouns or proper nouns by the model. This was resolved by looking for a digit in the string.
- (5) All symbols are assigned the symbol (SYM) tag.
- (6) WALA tag is applied for all wala words.

The model with heuristic rules applied shows improvement in the accuracies of VBL, VB, VBI, and AUXT tags. NN and NNCM are also marked with higher accuracy. These improvements, though minor, impact the performance of NE tagging (see Section 9).

NE tagging is also affected by the shallow parsing information (see Section 9.1.1). Hence the next step in our NLP pipeline is to provide shallow parsing information for Urdu data.

# 7. SHALLOW PARSER

Shallow parsing gives non-overlapping phrase boundaries that are used to identify the phrase structure in a sentence. Algorithms for shallow parsing use syntactic rules to perform detailed semantic analysis such as role labeling [Punyakanok et al. 2005] and machine translation [Garrido-Alenda et al. 2004]. Linguistic analysis at the syntactic level is concerned with how words are arranged in a construction (sentence, phrase, etc.). The position of a word and its role in a construction offers great insight in understanding the meaning of the construction.

Very little work has been done to provide shallow parsed information for a language such as Urdu. Rizvi et al. [2004] developed a rule based shallow parser (or chunker) for Urdu using the linguistic characteristics of morphologically closed classes. Their method uses heuristic rules that group chunks based on the closed class words. The linguistic nature of the closed class words influence the joining of neighboring words to these chunk groups. They also

Table IV. Results of MaxEnt-Based Urdu Tagger Before and After Using Lexicon Lookups and Heuristic Rules

Column 1			Colu	mn 2			Colu	mn 3	
Tag	POS	ACT	PREC	REC	F	ACT	PREC	REC	F
AUXA	275	253	0.81	0.745	0.777	248	0.851	0.767	0.807
AUXT	475	417	0.93	0.817	0.87	430	0.937	0.848	0.891
CC	226	229	0.978	0.991	0.985	227	0.982	0.987	0.985
CD	539	444	0.89	0.733	0.804	495	0.901	0.827	0.863
CM	2149	2141	0.992	0.988	0.99	2131	0.996	0.987	0.992
DATE	12	8	0.75	0.5	0.6	11	0.909	0.833	0.87
DM	64	40	0.475	0.297	0.365	60	0.45	0.422	0.435
DMRL	24	1	0	0	0	1	0	0	0
FR	11	3	1	0.273	0.429	8	1	0.727	0.842
I	68	31	0.774	0.353	0.485	25	0.92	0.338	0.495
INJ	2	0	0	0	0	0	0	0	0
ITRP	28	26	1	0.929	0.963	27	1	0.964	0.982
JJ	1207	1162	0.746	0.718	0.732	1181	0.767	0.751	0.759
JJRP	6	8	0.625	0.833	0.714	8	0.625	0.833	0.714
KER	4	5	0	0	0	1	1	0.25	0.4
MOPO	14	1	1	0.071	0.133	4	1	0.286	0.444
NN	2518	3058	0.747	0.907	0.82	2920	0.791	0.917	0.85
NNC	60	40	0.65	0.433	0.52	33	0.758	0.417	0.538
NNCM	331	371	0.86	0.964	0.909	368	0.872	0.97	0.918
NNCR	70	18	0.833	0.214	0.341	24	0.792	0.271	0.404
NNP	603	636	0.594	0.627	0.61	564	0.67	0.627	0.648
NNPC	490	369	0.818	0.616	0.703	410	0.751	0.629	0.684
OD	36	19	0.842	0.444	0.582	18	0.889	0.444	0.593
PM	841	833	0.978	0.969	0.974	797	0.981	0.93	0.955
PR	295	291	0.77	0.759	0.765	282	0.794	0.759	0.776
PRP\$	2	0	0	0	0	0	0	0	0
PRRF	4	2	1	0.5	0.667	0	0	0	0
PRRFP\$	45	43	0.977	0.933	0.955	44	1	0.978	0.989
PRRL	80	102	0.775	0.988	0.868	101	0.772	0.975	0.862
Q	147	0	0	0	0	161	0.64	0.701	0.669
QW	6	4	0.75	0.5	0.6	4	1	0.667	0.8
R	7	0	0	0	0	3	1	0.429	0.6
RB	153	140	0.843	0.771	0.805	132	0.924	0.797	0.856
RBRP	7	5	0.4	0.286	0.333	4	0.5	0.286	0.364
SC	184	187	0.952	0.967	0.96	186	0.962	0.973	0.968
SM	461	457	0.978	0.97	0.974	504	0.893	0.976	0.933
SYM	129	34	1	0.264	0.417	132	0.977	1	0.989
U	4	0	0	0	0	1	1	0.25	0.4
VB	534	555	0.69	0.717	0.703	556	0.719	0.749	0.734
VBI	117	108	0.648	0.598	0.622	118	0.644	0.65	0.647
VBL	193	203	0.409	0.43	0.419	195	0.451	0.456	0.454
VBLI	116	83	0.831	0.595	0.693	81	0.84	0.586	0.69
VBT	70	132	0.386	0.729	0.505	122	0.443	0.771	0.562
WALA	75	73	1	0.973	0.986	75	1	1	1

Column 1: POS tag names and the number of instances for each tag, Column 2: Results with only word level features, Column 3: Results with word level features + lexicon lookups + heuristic rules, ACT: number of instances found by the model, PREC: Precision, REC: Recall, F: F-Measure (%).

provide a complete parser for Urdu which is rule based and builds on the chunks that are formed. This method has not been quantitatively evaluated to determine its effectiveness. However, considering the complexity of the language and the dependency of the method on heuristic rules, formulating a complete unambiguous set of linguistic rules that cover the entire language is very challenging and involves considerable amount of effort. In the literature, shallow parsing using statistical techniques has also shown promising results. Techniques based on HMMs and CRFs have been very successful in generating shallow parsers for other languages like Chinese [Liao et al. 2006] and Hindi [Awasthi et al. 2006]. To the best of our knowledge, at the time of writing this article, there is no data freely available which is annotated with chunk information for Urdu. Since developing a statistical learning model requires access to annotated data, alternate techniques based on resource sharing is considered.

Each of the chunks in the shallow parser is represented by chunk tags. These chunk tags are language dependent as they depend on the syntactic structure of the language. Chunk tags, though not exclusively designed for Urdu, have been formalized for Indian languages as a part of the AnnCorra standards [Bharati et al. 2006]. A chunk tag depends less on a word itself and more on the POS category associated with the word. Since Hindi and Urdu share a similar syntactic structure, borrowing Hindi tags for use with Urdu seems logical and promising. This idea of adapting resources from a language that is more privileged to a language that is less privileged is not new. Singh and Surana [2007] conducted studies that help in estimating the cost involved in borrowing resources. They show that, based on the task involved and the kinds of similarities considered between the languages, it is acceptable to depend on a linguistically close and resource rich language for analysis. Singh and Surana [2007] proposed different methods for comparing similarities between languages. Their methods calculate the surface similarity, contextual similarity, and distributional similarity to compare similar languages.

Singh [2006] proposes various distance measures to identify language encoding based on language similarities. The measures they consider range from simple log probability difference to more complicated measures like the JC measure [Jiang and Conrath 1997]. We conducted a simple experiment to justify the syntactic similarity between Hindi and Urdu. To compare only the syntactic similarity (not the vocabulary), the RE measure [Singh 2006] is sufficient as we need to measure only the relative entropy between the POS transition probability distributions. Four different corpora, each with approximately 50,000 words—two for Hindi, one for Urdu and one for English—are used. The corpora are tagged using a condensed tagset that marks nouns, adjectives, pronouns, adverbs, verbs, conjunctions, interjections, and prepositions. The transition probability of these syntactic categories for each corpus is calculated. Words that could not be assigned to any of the above mentioned syntactic categories are assigned a probability of 1/count(POS tags) as such words influence the syntactic difference between the languages.

Figure 1 shows the KL divergence scores calculated for 3-, 4-, and 5-gram POS transitions. We see that the syntactic divergence between Hindi and

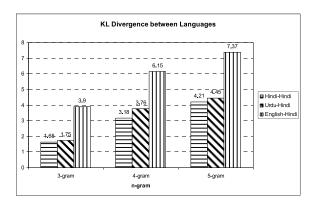


Fig. 1. KL divergence between languages showing syntactic similarity.

Urdu is only slightly greater than the syntactic divergence between Hindi and Hindi. However, the syntactic divergence between Hindi and English is much greater than that between Hindi and Hindi. We can now conclude that Urdu and Hindi have very similar syntactic structure and the chunk tags from Hindi can safely be borrowed over to Urdu. Note that this experiment only justifies the syntactic similarity between Hindi and Urdu and not the similarity over the vocabularies.

In any language the construction of a sentence is a combination of two kinds of words: open class and closed class. Open class words, or content words, such as nouns, verbs and adjectives, are the main bearers of meaning in a language. They contain crucial semantic information and provide the building blocks needed to comprehend the overall sense of what is spoken or written in a sentence. Closed class or function words such as articles, conjunctions, and prepositions support syntactic analysis of a sentence. These words are relatively devoid of meaning and primarily serve a syntactic role in language understanding. They co-determine the syntactic relations between open class words, thereby making word combinations interpretable. We can consider this distinction between open class and closed class words as a basic reflection of the separation between semantics and syntax. This distinction motivated us to consider only the role of closed class words in proximity to POS categories of open class words in generating an effective Urdu specific chunker.

# 7.1 Using Hindi Chunker for Urdu

The tagset for chunking as defined by the AnnCorra standards is show in Table V. An obvious approach is to apply a Hindi specific shallow parser on Urdu data. A Hindi chunker with five-fold cross-validation accuracy of 91.2% was trained using CRF learning approach on the POS and chunk annotated data provided for Hindi (IIIT Hyderabad corpus). The annotated data set consisting of about 20,000 words were divided into training and testing sets (approximately 15,000 and 4,000 words, respectively). Features used for training are current word, current POS tag, bigram word, and POS tag probabilities.

Table V. Tagset for Chunking as Defined in the AnCorra Standards

Chunk Tags (SP Tags)			
Noun chunk	NP	Verb Chunk (Gerund)	VGNN
Finite Verb chunk	VGF	Adjectival Chunk	JJP
Non-Finite Verb chunk	VGNF	Adverb Chunk	RBP
Infinitival Verb chunk	VGINF	Chunk Fragments	FRAGP
Chunk for Negatives	NEGP	Miscellaneous	BLK
Conjuncts	CCP		

Table VI. Chunk Tag Assignments for Example 1 Using Hindi Chunker and Reduced Chunker

English	Act	ual Tags		tags with		nunk tags
transliteration			Hindi tagger		with reduced	
			or	urdu Urdu	corp	us training
	(	NP	(	NP	(	NP
Miss	مس باگ	NNP	مس	NNP	UNK	NNP
Haag	ہاگ	NNPC	ہاگ	NNPC	UNK	NNPC
	)		ايليانتي	NNP	)	
	(	NP	)		(	NP
Elianti	ايليانتي	NNP	(	NP	UNK	NNP
ka	کا	$\mathbf{C}\mathbf{M}$	کا	$\mathbf{C}\mathbf{M}$	کا	$^{\mathrm{CM}}$
	)		)		)	
	(	NP	(	VGNN	(	NP
kirdar	کردار	NN	کردار	NN	UNK	NN
	)		)		)	
	(	VGF	(	VGNF	(	VGF
ada	ادا	VB	ادا	VB	UNK	VB
karti	کرتی	VBL	کرتی	$_{ m BL}$	کرتی	VBL
hai	ہے	AUXT	ہے	AUXT	ہے	AUXT
	-	$_{\mathrm{SM}}$	-	$_{\rm SM}$	-	$_{ m SM}$
	)		)		)	

When this chunker was applied on Urdu data (with transliterations similar to what is done in Section 6), the results were not impressive. Unfortunately, due to the lack of gold-standard annotated data for Urdu, the results of this chunker cannot be quantified for Urdu. However, example 1 and its explanation in Table VI should give the reader a fair idea of the issues associated with applying a Hindi chunker on Urdu data.

Example 1. - مس باگ ایلیانتی کا کردار ادا کرتی ہے (Miss Haag Elianti ka kirdar ada karti hai).

Column 2 of Table VI shows the actual assignment of chunk tags for Example 1. Column 3 of Table VI shows the assignment of chunk tags obtained by applying a Hindi chunker on Urdu for Example 1. The discrepencies are (see Section 6.1) due to the vocabulary difference between Hindi and Urdu. Since all the word tokens were also considered as features while training the learning model, several of the Urdu specific words are considered unknown words and the chunk boundaries are not marked accurately. Several of the chunks that need to be marked NP (noun chunk) are marked VGNN (gerunds) in the context of a noun POS tag followed by a verb POS tag (kirdar and ada). Chunk

boundaries for proper nouns are also not accurate. These errors indicate that the Hindi chunker has to be modified to deal with Urdu related anomalies.

# 7.2 Urdu-Specific Chunker

Chunking is highly dependent on the syntactic structure of a sentence. Closed-class words, as mentioned before, support the syntactic analysis of a sentence. Intuitively, training a learning model with only the closed class words for chunking should yield good results. Both Rizvi et al. [2004] and Stanislav [2003] state the importance of closed-class words over open-class words for the purpose of chunking. Stanislav [2003] shows that the words that commonly match between Urdu and Hindi are closed-class words. We developed a shallow parser by training a CRF learning model on a reduced corpus of Hindi shallow parsed annotated data. This model is then applied on Urdu to generate the chunk tags.

A reduced Hindi corpus consisting of words that belong to the closed class—case markers, postpositions, possession markers, conjunctions, interjections, auxiliaries, pronouns, and question markers—is created. Only the training set of the annotated data is reduced to this format. The test set is kept in its original format. We train a CRF learning model which considers word token features only if the token belongs to the closed class group. The bigram probabilities of word tokens around the closed class words and the POS tags associated with them are also considered as features when training the model. The five-fold cross-validation accuracy of this model is 89.99% which is close to the accuracy obtained (91.2%) when the features of open class words are also considered for training, thereby indicating the potential of this technique (see Column 3 of Table VI).

To make this technique more compatible for Urdu in the nastaliq script, the closed-class words in the reduced corpus were translated to Urdu using a Hindi-Urdu dictionary (in most cases, this process is a simple transliteration as closed-class words in Urdu are the same as in Hindi [Stanislav 2003]). However, there are some words that require a dictionary lookup. For example, Hindi word "sriman" to Urdu word "janab". This was not a difficult task as the set of closed-class words is limited and hence the replacements were minimal. Another CRF learning model was trained on this set to generate an Urdu specific chunker. We perform a task based evaluation of this Urdu specific chunker and show that the F-measure of an NE tagger (see Section 9) improves from 71.4% to 73% when these chunk tags are used as features in training the NE tagger (see Section 9.1.1).

## 8. MORPHOLOGICAL ANALYZER

Morphological analysis of words is a process in which words are studied for their structure based on the different forms that they represent. Usually, morphological analysis is performed to aid POS tagging [Agi et al. 2008]. However one cannot rule out the usefulness of detailed morphological analysis for more advanced NLP tasks like machine translation [Stymne et al. 2008]. The

CRULP POS tagset used by us (in Section 6) is fairly comprehensive. It covers all the top level POS categories including more detailed analysis for verbs (light verb, infinitive verb) and nouns (prepositional noun, combined noun). This tagset does not account for gender, case, number, and tense inflections for nouns, verbs, and adjectives. Information about these inflections, if present, can benefit several tasks like NE tagging and semantic role labeling. In fact, some of the heuristic rules applied during the post processing stage of the four stage NE tagger that we developed for Urdu (Mukund and Srihari [2009] and Section 9.1) use the gender information of nouns for location entity correction. Several bootstrap techniques and transfer learning techniques used for resource poor languages can effectively use inflection information (e.g., attempts to generate annotated data for resource poor languages using cross-lingual projections can yield promising results if detailed morphological inflections over verbs are considered). These benefits of detailed morphological analysis motivated us to integrate one such module to our NLP pipeline.

Performing morphological analysis in Urdu involves identifying the different lexical categories to which a word belongs, along with formulating rules that categorize these words to their respective lexical categories. Urdu and Hindi have very similar grammatical structure. However, there are a few significant differences that prevent the usage of Hindi morphological analyzer for Urdu. Following are some of the differences.

- —The phonology, vocabulary, and writing styles between the two languages differ.
- —Postpositions are treated as bound morphemes after pronouns in Hindi, but as separate words in Urdu (Wikipedia Urdu Grammar page<sup>19</sup>).
- —Variations associated with the third person form are another grammatical difference between Hindi and Urdu. yah "this"/ye "these"/vah "that"/ve "those" is the literary set for Hindi while ye "this, these"/vo "that, those" is the set for Urdu and spoken (and also often written) Hindi (Wapedia Urdu Grammar, page 12).

Bögel et al. [2007] have developed a morphological analyzer for Urdu as part of the Urdu ParGram project [Butt and King 2002]. Humayoun et al. [2007] have also performed Urdu morphological analysis using functional morphology. Both methods begin with an attempt to determine the top level POS category (noun, verb, adjective, preposition, adverb) and later, based on heuristic rules, account for inflections. However, both analyzers have not been quantitatively evaluated. Providing a quantitative performance measure for a morphological analyzer is a hard task as manually annotating a corpus that accommodates all inflectional variations is difficult. Since both methods are rule based, qualitative analysis of the rules helps to determine the performance of the analyzers. Humayoun et al. [2007] documented all the rules used to mark the top level tags and the inflections associated with them. However, these rules produce multiple categories for each word and hence multiple inflections which require

<sup>&</sup>lt;sup>19</sup>See http://en.wikipedia.org/wiki/Hindi-Urdu\_grammar.

a separate model to perform disambiguation. The quality of these rules is superior as they are modeled based on the Urdu grammar analysis done by well known grammarians, Butt and King [2004] and Siddiqi [1971]. We used a variation of the work of Humayoun et al. [2007] to develop our Urdu morphological analyzer. In our approach, the lexical categories are obtained by first running the statistical POS tagger (see Section 6) where suitable POS tags are assigned to the words. Our analyzer is only used to determine the inflections for words that already have POS labels. As our morphological analyzer is also rule based errors, if present, will be those propagated from the POS tagger.

Semantex<sup>TM</sup> has developed its own grammar formalism and implementation. The underlying automaton corresponds to a tree walking automata. The rules however are regular expressions on the tokenlist (see Section 3.1). The grammar rules are written by considering token descriptions and combining them with regular expressions. Token descriptions are collections of constraints that individual tokens must satisfy where each word is represented by a token. The basic operations in a rule are (1) tagging: POS tagging and NE tagging, (2) chunking: shallow parsing, and (3) linking: relationship identification.

 $Example \ 2. \ [:: [\p{Alphabetic}]+پر+b::/t \ (NNP \mid NNPC) \sim NeOrg \sim NePer < NeLoc>]$ 

The rule in Example 2 is used to change the NE tag of all those words that are marked proper nouns (NNP or NNPC) and have suffix ending with "\*x" (pur). ~NeOrg and ~NePer imply the removal of these tags (if they exist on the word) and assigns <NeLoc> tag to the word. (Words such as Nagpur, Jaipur, etc., are tagged NNP and have suffix ending "pur". These are considered Location Entities).

Srihari et al. [2008] discuss the architecture and the grammar framework of Semantex<sup>TM</sup> in detail. We have used the grammar toolkit of Semantex<sup>TM</sup> to develop the morphological analyzer for Urdu. The advantage of using this toolkit over other toolkits (XFST [Xerox Finite State Tool]) is that our analyzer can be seamlessly integrated into the Urdu NLP pipeline framework that also uses the Semantex<sup>TM</sup> platform. Humayoun et al. [2007] provide rules for determining nouns, verbs, adjectives, adverbs, and pronoun inflections. In the following subsections, we briefly explain the different inflections considered by Humayoun et al. [2007] in context of our implementation.

## 8.1 Nouns

Most nouns are identified based on post-positions (clitics). Nouns are inflected in number and case. They can be singular or plural. They also inflect on gender. The Urdu POS tagger developed by us (see Section 6) uses the CRULP tagset that successfully identifies four variations of nouns: noun (e.g., لاك ladka  $\sim$  boy), combined noun (براه راست) bara-i-rast  $\sim$  direct), prepositional noun ( $\sim$  and ar  $\sim$  inside), and proper noun with an F-measure of  $\sim$ 70%. We use heuristic rules to mark the various inflections on these nouns. Humayoun et al. [2007]

Table VII. Urdu Cases

Case	Clitic Form with Pronunciation	Morphological Effect	Examples
Nominative Nominative		Nominative or no change	
Oblique	-	Nominative or its modified form	
Ergative	نے (ne)	Oblique + ne	الی نے کیتا پدھا (Ali ne kitab) pada ~ Ali read a book)
Accusative	(ko) کو	Oblique + ko	ا <b>لی کو</b> کیتاب دیا (Ali ko kitab diya) ~ Gave the book to Ali)
Dative	(ko,ke) کے , کو	Oblique + (ko, ke)	Similar to accusative
Instrumental	(se)	Oblique + se	قلم سے لکھا (qlam se likha ~ wrote with a pen)
Genitive	کی <sub>ب</sub> کے <sub>ب</sub> کا (ka, ke, ki)	Oblique + (ka, ke, ki)	الٰی کا کیتاب ہے (Ali ka kitab hai (~ the book is Ali's)
Locative	تلے , تک , پر , میں , (main, par, tak, tale, talak)	Oblique + (main,par, tak, tale, talak)	گهر میے کیتاب هے  Ghar main  kitab hai ~ the book  is in the house)
Vocative	(ae) آے	ae + Oblique or modified form of Oblique	ل <b>ڑکو!</b> سنو (larkho! suno ~ boys listen!)

consider nine cases for nouns: nominative, oblique, ergative, accusative, dative, instrumental, genitive, locative, and vocative. All nine cases are identified based on the clitic information. Each case (Table VII) is programmed as a rule and applied over each word token identified as a noun by the POS tagger.

There are two genders in Urdu: masculine and feminine. Table VIII shows how genders are identified in nouns based on letter endings. For each of the rows in the table, case information is marked based on the clitics. For a complete list please refer Humayoun et al. [2007].

# 8.2 Adjectives

Humayoun et al. [2007] account for two types of inflections in adjectives. One type of adjective inflects in case, number, and gender and the other inflects in degree (positive, comparative, and superlative). Gender information is taken into account to mark the first type of inflection in adjectives. No inflections are associated with the masculine form of adjectives if they do not end with (alif), for example,  $\dot{\epsilon}$  (khoobsurat  $\sim$  beautiful). Feminine forms generally end with  $\epsilon$  (choti ye), for example,  $\dot{\epsilon}$  (sultani  $\sim$  monarchic). Masculine forms end with (alif) or  $\dot{\epsilon}$  (bari ye), for example,  $\dot{\epsilon}$  (harabara  $\sim$  magnificent). In order to identify degree inflections, the phrase used before the adjective plays an important role. Adjectives with a positive degree do not have any

Index Case Conversion Plural nominative/Singular oblique – if last letter is Singular Masculine (کے bari ye) e.g. کڑکے ~ boys nouns ending with الرُّكُون .Plural oblique – if last letter is (عور un) e.g. (Ialif,  $\circ$  choti he,  $\xi$  aen) الركو (va) e.g. الركو Plural vocative – if last letter is Singular Masculine Plural nominative/Singular oblique – last occurrence nouns ending with of (\( laif \)) replaced by (\( choti ye \)) (u / an, aN)Plural oblique/Plural vocative – last occurrence of (/alif) replaced by (ugun) e.g. کنو اس کنو to کنوی ( $well\ to\ wells$ ) 3 Singular Feminine Plural nominative – added  $(\sqrt{an})$ nouns ending with Plural vocative – added ( $\mathcal{J}va$ ) Plural oblique – added  $(\mathcal{O}un)$ (\(\mathcal{S}\) choti ye, \(y\)) e.g. لڑکیاں to لڑکی (girl to girls) 4 Singular Feminine Plural nominative – ends with (بر yeN) Plural vocative - ends with  $(\vec{9} \ vao)$  or  $(\vec{0} \ an)$ nouns ending with Plural oblique – added ( $\tilde{e}$  on) (lalif, Ulan, (Uson))e.g. مانیس to ماییس (mother to mothers) Singular Feminine Plural nominative – ends with (\(\mu nun gunna\)) nouns ending with Plural vocative – ends with ( ¿va)  $(\underset{\boldsymbol{\mu}}{\boldsymbol{\mu}} ya)$ Plural oblique – ends with (u)une.g. گڈیاں to گڈیا (doll to dolls) Singular Feminine Plural nominative – ends with (پيرو) nouns not ending with Plural oblique – ends with ( \( \( \sup \) \( \sup \) un) ( 'Jalif, ∪nun gunna, Plural oblique – ends with  $(\mathfrak{g}va)$ yvao, on, wN)e.g. جورو $\sim wife$ Singular Masculine Plural nominative – added (a)nouns ending with Plural vocative – added (  $\mathcal{J}va$ ) Plural oblique – ends with (علات) (عالت) (عالت) (inun, ار)ar) e.g. احسانو to lavour to favours) Singular Feminine Plural nominative – added (עש, אין) nouns ending with Plural vocative – ends with ( ¿va)

Table VIII. Gender Information for Nouns

intensifiers as affix phrases. However, if the word is preceded by phrases like (bohot se, thoda zyada etc.) then the degree is comparative. Affix phrases such as (sab se, bohot zyada etc.) determine the superlative form of the word. The rules that help to mark these inflections are applied on the JJ (adjective) tagged words only.

Plural oblique – ends with ( $\upsilon un$ ) e.g. محبتیں to محبتیں (love to love)

## 8.3 Verbs

(*نےte*)

Humayoun et al. [2007] consider tense, mood, aspect, gender, and number inflections of verbs and the rules take all of these factors into account. In our implementation we consider only the tense information. The rules used by our analyzer to identify verb inflections are shown in Table IX.

Verbs are also categorized as transitive, intransitive, and causative verbs. The POS tagset that we use has two infinitive tags for verbs: VBI and VBLI.

Table IX. Tense Information for Verbs

Present	Verbs and Light verbs followed by Auxiliary verbs ending with (', , , , )  "ha/hey/hey" indicate present tense. But the tense of such words can also indicate future based on the actual auxiliary verbs that follow them. Hence, verbs are first checked for present tense and later followed by the ones mentioned below. e.g. padte hey (will study), pad rahe hey (are studying)
Future	Auxiliary verbs like (گے کی گی "ga"/"ge" preceded by (بین/بوں) "ho"/"hon" indicate future tense with their respective gender information being masculine/feminine/masculine e.g. baarish ho gi (it will rain)
Past	The past tense of a verb is indicated by the presence of adverbs like ((ع) "ho/hon" followed by aspectual auxiliary verbs ending with (ع) "ye". Also the presence of tense auxiliary verbs like (أنَّ عَلَى اللهُ عَلَى اللهُ اللهُ اللهُ عَلَى اللهُ اللهُ عَلَى اللهُ
Imperfective	Light verbs or verbs like (אָפנדים אָפְפּדין) "hota/hote/hotey" and "kraz" indicate imperfective tense. (צ'דים אַצ'דין) "lata/late/latey" indicate imperfective tense. e.g kitab latey they (was getting the book)
Perfective	Auxiliary verbs like (أبوئس لبوئس "hua/hui/hue" indicate events of the past but give a transition to either the present or the future. Such words indicate perfective tense. e.g. wo ghussa hua (he got angry)

Table X. Rules for Verb Classification

1	Most infinitive verbs that end with "nna" indicate either infinitive transitive or
	intransitive verbs with words that end with "nne" to be oblique. e.g. banna (become)
2	Most infinitive verbs that end with "nana" indicate infinitive direct causative and
	"nane" indicate oblique direct causative. e.g. banana (to make, to become)
3	Most infinitive verbs that end with "wana" indicate infinitive indirect causative
	and "wane" oblique indirect causative.
	e.g. banwana (to get done)

VBI is used to indicate the infinitive form of verbs and VBLI to indicate the infinitive form of light verbs: These two tags play a very important role in marking infinitive inflections for verbs (Table X).

# 8.4 Pronouns

Pronouns inflect in number, person, gender, and case. But these inflections are not regular and sometimes may show no inflections. Hence developing generalized rules for pronoun inflections is difficult. But Humayoun et al. [2007] have compiled case by case rules to handle pronoun inflections based on the most common pronouns.

 Categories
 Instances

 Location
 1262

 Organization
 1258

 Person
 1772

Table XI. NE Statistics for CRL Annotated Corpus

relation. Case inflections for each of these pronouns are determined based on the word endings and case markers that follow. The details can be found in Appendix A of Humayoun [2006]. Person information for these pronouns plays a very significant role in anaphoric resolution.

Since all the rules are borrowed from Humayoun et al. [2007], we tested our system by applying our analyzer on the examples mentioned in their work. There are 30 example sentences for noun inflections. Our analyzer successfully marked case and gender inflections for all 30 sentences accurately. We propose to test our analyzer using a task based approach like semantic role labeling that will eventually become part of the NLP pipeline framework that we are attempting for Urdu.

In the next section, we explain the steps taken to develop an Urdu NE tagger. The usefulness of our morphological analyzer can be seen in the heuristic rules used to improve the NE tagging accuracy (see Section 9.1).

#### NAMED ENTITY TAGGER

NE recognizers are an important tool for almost all types of NLP applications such as information extraction, summerization, machine translation, etc. NE tagging is a process of recognizing chunks that represent person, location, organization, date, time, percentage, or quantity. Accurately identifying these chunks is a nontrivial task. Several approaches, both rule-based and statistical learning-based have been used in the literature to provide high NE recognition (NER) rates across different languages. Successful approaches using machine learning techniques have been those that treat this problem as a sequence labeling task. MaxEnt [Borthwick 1999], HMM [Bikel et al. 1999], and CRF [McCallum and Li 2003] based methods are most popular. Ekbal and Bandyopadhyay [2010] have shown that using SVM based approach for Hindi NER also produces reasonably good results. There is limited work on NER for Urdu. The IJCNLP 2008, shared task for Named Entity Recognition has released an NE annotated data set for Urdu based on the AnnCorra standards [Bharati et al. 2006]. The best F-measure for Urdu NER on this data set is 35.47% reported by Saha et al. [2008] indicating the complexity of the task. The individual recognition rates for person and location categories are also very low. This can be due to the complexity of the tag set (which is very detailed) and the small size of the annotated data (only about 35,447 tokens). Unfortunately this is the only NE annotated data set freely available for Urdu. However, CRL has released an Urdu NE annotated corpus for the machine translation task. This corpus has a fairly simple tagset and is annotated for location, organization, person, and time categories. The corpus has about 55,000 words with a wide range of location, organization, and person names (Table XI).

Before exploring the usage of different learning models for Urdu on the above mentioned dataset, it is worthwhile to look at the NER performance for Hindi as the similarity between the two languages cannot be ignored. An NE recognizer for Hindi developed by Saha et al. [2008] has an *F*-measure of 79.03%. To overcome the problem of overfitting that occurs with using a MaxEnt classifier on a small data set, they use a clustering based approach to reduce the number of features. However the best performance obtained by using MaxEnt classifier is 75.6% *F*-measure. McCallum and Li [2003] also developed an NE tagger for Hindi using the CRF learning model with an F-Measure of 85.11%. These results show promise in using a learning model that treats NER as a sequence tagging problem.

Building an NE tagger for a language like Urdu is challenging for a number of reasons. For instance, Urdu does not have the concept of capitalization of characters. Also, most names of people have specific meanings associated with them and can easily be found in a dictionary as word entries. Our attempt to develop an NE recognizer for Urdu is explained in Mukund and Srihari [2009]. Below, we give a brief explanation of this technique and also continue to explain further enhancements made (adding chunk features and using the ACE tagset) to improve the performance of the model.

The first step was to establish a baseline score. We use a MaxEnt classifier. We decided to use the POS information associated with each word as a feature, as POS tags roughly account for the sense of the word. We consider three main categories: location, organization, and person. A 10-fold cross-validation on the CRL data set using this MaxEnt classifier has an *F*-Measure of 55.3%. It is obvious that all these NE tagged words should be tagged with the NNP (proper noun) POS tag. But when POS tags were generated for the NE tagged ground truth data, most of these words were either tagged as adjectives (JJ) or common nouns (NN). This adversely affected the NE tagger performance. We also notice that the POS tagger tagged most of the NNPs as NNs because of the sparseness of the NNP tag in the POS ground truth data set. This observation made us look at bootstrapping techniques for effective learning.

A four-stage model proposed by us in Mukund and Srihari [2009] uses a bootstrap technique to increase the accuracy with which the words are tagged as NNPs. Although we report results using CRF learning approach, MaxEnt approach also generates similar results. The working of this model is briefly explained as follows. The CRULP dataset (dataset<sub>POS</sub>) is a corpus of 150,000 words that are only POS tagged and the CRL dataset (dataset<sub>NE</sub>) is a corpus of 50,000 words that are only NE tagged. A POS learning model (see Section 6) is first applied on dataset<sub>NE</sub> to generate POS tags. Rules to correct the POS tags associated with the named entities is then applied to generate correct POS tags for dataset<sub>NE</sub>. Another POS learning model is trained on the combined dataset - dataset $_{NE}$  and dataset $_{POS}$ . This model is now used to generate POS tags that are used as features to learn the NE tags. The final NE learning model is trained on dataset<sub>NE</sub> with the corrected POS tag information. More details of this algorithm can be found in Mukund and Srihari [2009]. A ten fold cross validation on this model has an average F-measure of 68.89% with the maximum being 69.21%.

This technique of bootstrapping gave us two advantages:

- (1) an increase in the size of the dataset used to train the POS learning model,
- (2) more NNP entries that enable better learning of proper noun POS tags.

The CRL data is newswire material, and these articles are written in the "journalistic" or "news writing" style. <sup>20</sup> The articles are objective and follow a Subject-Object-Verb structure. Related information is usually presented within close sentence proximity. This makes it possible to hand-craft grammar rules for the discovery of NE tags with fine granularity. The final POS tagged and NE tagged data generated is processed using rules and lexicon lookups to further improve the overall tagging accuracy of the model. The rules used in this method are domain specific.

# 9.1 Rules for NE Tagging

- (1) Words such as "غورت" (court), "بيورو" (bureau), and "فوع" (army), etc., are looked up. If there are any nouns or proper nouns preceding these within a window of two, then the tag on this word is ORGANIZATION.
- (2) Words such as "آرمی" (organization), "آرمی" are marked ORGANIZATION if the previous word is a proper noun.
- (3) Lexicon look up for names of places is performed and the POS tag of the next found word is checked. If this tag is a Case Marker (CM) with a feminine gender, like "كے" (main) or "مين", then the word is marked with a LOCATION tag.
- (4) If a proper noun that is selected ends with a suffix "pur", "bad, "dad" and has the same constraint as mentioned in Rule 3, then the LOCATION tag is assigned to it as well.

Applying these heuristic rules improved the *F*-measure for the worst test set from 68.89% to 74.67% and for the best from 69.21% to 71.3%. Our test sets consist of about 100 tags from each category. We also trained a MaxEnt learning model on a combined data set: CRL data set and the data set released by IJCNLP (2008). The *F*-measure on this data set was 34.2%, much less than the baseline of 55.3%. This could be due to the domain difference between the two data sets and also the styles of writing. Since using MaxEnt on CRL data set alone showed more promise, we proceeded with using only this data set for our experiments.

9.1.1 *Chunk Information for NE Tagging*. Using chunk information to improve NE recognition is not new. Several participants who achieved state-of-the-art NER scores for English in MUC-6 and MUC-7<sup>21</sup> NE shared tasks used chunk information [Zhou and Su 2002]. To further improve the efficiency of the proposed four stage model, we decided to use the shallow parsed information along with the parts of speech in the NE learning phase. The shallow tags

 $<sup>^{20} \</sup>rm See~http://en.wikipedia.org/wiki/News\_writing.$ 

<sup>&</sup>lt;sup>21</sup>See http://www-nlpir.nist.gov/related\_projects/muc/proceedings/muc\_7\_toc.html.

Table XII. NE Results for Four-Stage Model with and without Chunk Tags

Features	Tag	F-Measure (%)	F-Measure (%)
(w - word and pos		without Chunk tags	with Chunk tags
- part of speech)		(71.3% overall)	(73.1% overall)
$w_i, w_{i-1}, w_{i+1}, pos_i pos_{i-1},$	NeLoc (Location)	69.2	73.5
$pos_{i+1}$ , $chunk_{i-1}chunk_{i+1}$ ,	NeOrg		
pos bigrams. word bigrams,	(Organization)	72.5	72.8
lexical features	NePer (Person)	71.8	72.6

Table XIII. Subcategories for Location NE Tag in the ACE Tagset

<neloc></neloc>	GPE_Continent, GPE_Nation, GPE_Population_Center, LOC_Celestial,
(Location)	LOC_Land_Region_Natural

were generated using the technique described in Section 7. The *F*-measure on the test set improved from 71.3% to 73.1% as shown in Table XII. Including chunk information shows a significant improvement in the location recognition rate. This is because several of the location tags consist of multiple words, that is, each location tag chunk can contain more than one word. For example, Andra Pradesh and South Africa contain two words in the location chunk. Though our POS tagset accommodates for a Proper Noun Continue tag (NNPC), several of these were not marked accordingly. But the shallow parsed chunk tags accommodated for this error explaining the boost in the location recognition rate.

# 9.2 NE Tagging Based on ACE Tagset

Tasks such as machine translation and information extraction benefit from fine-grained categorization of NE tagsets. ACE [2005] NE tagset is a comprehensive tagset that provides several NE categories. In order to achieve better clarity, we decided to represent the location NE tag in the ACE format. Table XIII shows the various subcategories under which location words can be represented.

Both the test and training set used for NE tagging are now modified to use the sub categories for location tags. The location keywords are manually looked up in lexicons (see Section 3) to determine the subcategories. Proper nouns which do not occur in the lexicons or words that occur in more than one subcategory are further analyzed taking the surrounding context into account to determine the appropriate subcategory assignment. Words that cannot be assigned to any of the ACE tags retain the location (<NeLoc>) tag. A similar four-stage model is used to train an NE system using the modified annotated data. The learning model is a MaxEnt classifier with the same set of features as shown in Table XII. This classifier is applied on the fold with *F*-measure 73.1% (see Table XII). The performance of the classifier is reduced to 72.6% (Mean Precision = 75.0% Recall = 70.3%) (see Table XIV) when trained on ACE tagset. This reduction is due to the small variation triggered in the person categorization on account of similar syntactic structure with the location category.

A number of avenues remain to be explored to further improve the performance of the NE model. One approach would be to use the bootstrapping technique for NE data as well. However, the rules required can be complicated.

PREC REC Tags F GPE\_Continent 0.667 1 0.8 GPE\_Nation 0.659 0.885 0.755 GPE\_Population\_Center 0.727 0.762 0.8 NeLoc 0.688 0.917 0.786 NeOrg 0.794 0.706 0.748 NePer 0.827 0.615 0.706

Table XIV. NE Tagging Results for Some Tags in the ACE Tagset

PREC: precision, REC: recall, F: F-Measure (%).

More hand-crafted rules and detailed lexicon lookups can result in better NE tagging. Rules that resolve the ambiguity between location and person tags need to be explored. Since POS tags are considered as features, we notice that a lot of proper nouns are still getting tagged as adjectives adding to the error. Resolving this issue will definitely boost the NE tagging accuracy. We have explored the gender information of nouns and verbs that help in better location tagging. Exploiting similar morphological information such as the case information (locative, dative etc. (see Section 8) can also provide improvements.

## 9.3 Time Date Identification and Conversion

Since we do not account for the date information in the NE tagging model, a separate process had to be considered to identify date words. Identifying time and date information in Urdu text is a challenge. There are two main formats in which this information is represented.

- (1) Gregorian format that follows the English calendar.
- (2) Islamic format that follows the lunar calendar (Hijri calendar).

Urdu has specific markers that help to distinguish between the two formats. "a" (U+06BE) when used at the end or the beginning of a date representation indicates the Islamic format and "e" (U+0621) or nothing used at the end or the beginning of a date representation indicates the Gregorian format. Grammar rules are written to identify such representations and classify them accordingly. The rules also identify the different Gregorian formats: dd-mm-yy or dd/mm/yy. Our POS tagset also provides a date category tag and the POS tagger has a tagging accuracy of 90.9% on this tag (Table IV).

We also provide an algorithm that helps convert one date format into the other so that a date can be represented in either format. Refer the Fourmilab's Calendar Convertor page<sup>22</sup> for a detailed explanation of the algorithm used for conversion.

## 10. TRANSLITERATION

Transliteration is an important precursor to Machine Translation and Cross Lingual Information Retrieval (CLIR). It is a process of converting text written in one writing system into another writing system. The process is

<sup>&</sup>lt;sup>22</sup>See http://www.fourmilab.ch/documents/calendar/.

challenging when two writing systems having different orthographic styles are considered. In this section we describe a method that generates transliterations for common Urdu names while simultaneously restoring missing diacritics. We attempt to provide forward transliteration [Al-Onaizan and Knight. 2002] that involves transcribing an Islamic name written in Urdu to English. Our proposed hybrid method [Mukund and Srihari 2009] first learns the alignment probabilities over CV\* templatized words which are obtained by approximate transliterations, and then applies heuristic rules to obtain the best-transliterated words. The first level of character level transliterations applied in Mukund and Srihari [2009] were based on simple observations (observations are only done on the lexicons collected for common Urdu names (see Section 3) that resulted in 40% accuracy. We further conducted experiments that involved changing this first level character mappings (for ambiguous characters). Our results show that the mappings shown in Mukund and Srihari [2009] are the most accurate.

Performing named entity transliterations of Urdu words is a challenging task because of the issue of missing diacritics. While writing in Urdu, one tends to skip writing some vowels and this makes the script ambiguous. Based on the context and the restored diacritics, different words may look similar but have different associated meanings. For example "all," is both "Aalim", and "Aalam" based on the vowel that is restored between "l" and "m" and both have different meanings. There are also cases where an Urdu name has more than one correct transliteration. Such issues make the problem of transliteration very similar to that encountered for Arabic.

Stalls and Knight [1998] describe a transliteration process for Arabic that is based on a generative model. But their method requires all pronunciations to be documented and any out-of-vocabulary pronunciations will fail to generate the word sequence. Al-Onaizan and Knight. [2002] proposed a spelling-based model where each English letter is mapped to an Arabic letter in a sequence. This method out-performed the phonetic-based method. There are other methods that explore the information required for transliteration using parallel corpora. Samy et al. [2005] used Spanish to Arabic parallel corpora and picked the transliterated Named Entities based on sentence pair alignments. Kashani et al. [2007] proposed a three-phase algorithm for Arabic name transliteration that is based on a Hidden Markov Model approach. Their method also leverages lexical information to boost the overall accuracy of the system.

Methods that work well for Arabic name transliterations do not work well for Urdu. Urdu has a lot of aspirated consonants and retroflex sounds and these make pronunciations very different. Also the number of vowels between the two languages varies. Hence developing a unique Urdu transliteration system is beneficial. Diacritic restoration problems with respect to transliterations can be treated as a vowel restoration problem. If an Urdu word were to be transliterated character by character to English, then the learning problem becomes a problem of learning the possible vowels between the consonant transitions. Our proposed hybrid learning model is applied only if lexicon lookups fail to generate an English transliterated match for a given Urdu name. Refer to Section 3 for lexicon creation.

 Characters
 Hex
 Mappings Based on Observations

 U
 0X6BA
 n

 A
 0X6BE
 h

 C
 0X6CC
 y

 C
 0X6D2
 y

 J
 0X648
 w

Table XV. Ambiguous Character Mappings

A thorough inspection of our data set revealed the following inconsistencies. Some names starting with "a" [Aa] have their English transliterations starting with both "aa" and "a". While "aa" seems correct, "a" is not wrong either. Also the presence of aspirated consonants and "h" is not consistent. Urdu names that typically end with "a" [Heh goal] should have their English transliteration ending with "h". However there are many names that do not consider the presence of "h" in their transliterations. There are inconsistencies in the presence of diacritics as well. Some words include the diacritics and some do not. Treating these inconsistencies as errors is not accurate as several letters in Urdu do have dual pronunciations [Hussain 2004].

Our proposed hybrid approach is based on restoring the missing vowels. The first level involves representing the training data in a format that helps map the consonants between the two languages efficiently. This level has three major steps. In the first step, all of the Urdu names are approximately transliterated to English using the one-to-one Urdu to English character mappings (Table I in Section 3.1, Mukund and Srihari [2009]). These mappings were decided based on a thorough observation of the lexical resources collected for names. However, mappings used for characters shown in Table XV can trigger confusion.

Hussain [2004] explains the ambiguity associated with these characters. These characters can have more than one form of phoneme representation that depends mostly on their position in the word. Noon Gunna ("ω") is only for nasalization and should not be treated equivalent to "ن". Do chasmey Hey ("A") also has a similar issue which when combined with all stops and affricates [Hussain 2004] forms aspirated consonants but does not add additional phoneme [Hussain 2004]. These are true in the context of a text to speech system but not otherwise. But our observations reveal that several names with "ن" endings have their English transliterations ending with "n". For example, "shadan", "shadman", "tabaan" all end with "o". Same is the case with "a" which is equated to "h" in many cases. For example, "jahm", "wahbaan", "zahrah" all have "ه". "د" and "ك" have many English equivalents based on their position in the word. If they occur in the beginning and followed by only one phoneme, they are mostly equated with "j" otherwise "y". If they occur in the middle, they sometimes are "ae", "ai", "ei" or simply "y". We have considered "y" to begin with and later use heuristic rules that account for the position information to get the best transliteration. Similarly "," can be either "v", "w", "oo", "au", "ou".

In order to test our observations and the correctness of our initial character mappings, we changed these mappings to nothing for " $_{\circ}$ " and again nothing

Table XVI. Heuristic Rules for Approximate Transliterations

1.	If the current character	-If the previous character is the Arabic Kasra (zer - 0X650)		
	at the time of	and the character before this is the Arabic letter superscript		
	transliteration is "\"	Alef (0X6CC) then the current character is transliterated		
	(Alef - 0X627) and	to "aee".		
	positioned at the end	-If the previous character is the Arabic Damma (0X64F) and		
	of the word.	the character before it is "٤" (waw) then		
		the transliteration is "oo".		
		-If previous character is the Arabic Kasra (zer $-0X650$ ) then		
		transliteration is "ae".		
3.	If the current character is "\" - Arabic Alef with Madda above (0X622), then the			
	transliteration is "aa".			
4.	If the current character is the Arabic Kasra (zer – 0X650) and the next character is			
	a consonant then the tran	then the transliteration is "ee".		
6.	If the current character is the Arabic letter waw with Hamza above (0X624) and is			
	positioned at the end of the word, then the transliteration is "oo".			
8.	If the current character is the Arabic Kasra (zer – 0X650) and the previous is "ك"			
	Barree Yeh (0X6D2) then the transliteration is "ey".			
9.	If the current character is the Arabic Fatha (0X64E) and the previous is "ك" Barree Yeh			
	(0X6D2) then the transliteration is "ay".			

for "\$\sigma" if it occurred along with a consonant, "j" for "\$\sigma", "\$\sigma" if the word begins with either and is followed by a phoneme and "v" for "\$\sigma" (we also changed our heuristic rules to account for these characters). The results indicate that the character mappings based on our observations are the most accurate. Work done by Ahmed [2009] and the findings discussed in the paper complement our findings.

This step is followed by a set of heuristic rules needed to generate approximate transliterations. These rules help to mitigate the problem of inconsistent diacritic markings. The rules are generated based on the observations made on existing transliterations and by incorporating some mapping rules used by Ali and Ijaz [2009] in their English-to-Urdu transliteration system. Table XVI describes some of these rules. For a comprehensive coverage of these rules, refer to Section 3.1 of Mukund and Srihari [2009].

The next step involves generating consonant-vowel pairs. In our training data, both actual English transliterations and approximate transliterations generated in the first step need to have the same number of consonant-vowel pair mappings. A CV\* (C – Consonant V – Vowel) template is applied on every word and each pair is grouped separately. For example, "aabid" – "aabd" is split using the CV\* template to "aa bi d" – "aa b d". The inconsistencies present in the data required us to handle the templatization logic as two separate cases. In Case 1, there exists a one-one correspondence between the CV\* split for the actual transliteration and the approximate transliteration. In Case 2, such a one-one map does not exist. Hence correction symbols (^a and \$) are inserted into the actual English transliterations of such words. (see Section 3.1 of Mukund and Srihari [2009]).

The third step in generating the training data involves specifying the boundary conditions for each word pair. "B-" is added to the first character and "-E" is added to the last character of both the actual transliteration and the approximate transliteration word pair. Words for which the correction symbols

Table XVII. Heuristic Rules for Approximate Transliterations

- 1. Correct the word beginning with "aa" If the test word begins with "aa" and the found transliterated word begins with "a", then replace the first character of the found word with "aa".
- 2. Replace the first found "a" with "i"/"u" If the found word starts with "a" and not "aa". Verify the first character of the next four alignment strings. If the found word with a replaced "i"/"u" in the first character position occurs in the first four alignment strings, then this word is considered to be the most probable transliteration of the original Urdu name.
- 3. Replace "ai" with "aai" If the testword contains "aee" and is present in the middle of the word and the found word contains "ai" in the same position then the next four alignment strings are verified for the occurrence of "aai". If by replacing "ai" in the foundstring with "aai" makes it similar to the searched align string, then that string is considered to be the most probable transliterated string.
- 4. Prune repeated "i" at the end of the string If the test word ends with a "y" and the found word ends with "ii", then "ii" is replaced with "i" in the found word.
- 5. Correct "a" between consonants If "aa" is present between two consonants in the test word and the found word contains only "a" between the same two consonants, then replace this "a" with an "aa" if the word on replacement is found in the first four top alignment strings.

are applied are treated differently here as well. "B—" is appended to the first character of such word pairs.

## 10.1 Learning Model

The learning phase involves learning the possible vowels between consonants in a given word. We calculated the alignment probabilities using the generated training data as parallel corpora. These alignment scores are generated using GIZA++. <sup>23</sup> The similarity score between the pairs was determined by using the mapping table for potential CV\* alignments. The transliteration for a new Urdu name was found by calculating most probable CV\* alignments using beam search algorithm. The beam size is set to 75 and the alignment string with the highest probability was considered to be the most probable transliteration of the given Urdu name.

# 10.2 Post Processing Step

Heuristic rules based on the generated approximate transliteration are applied to move most transliterations with second highest alignment probabilities to being the most probable ones. These rules improve the accuracy of the system from 23% to about 60%. The rules (see Table XVII) account for the different variations associated with "3" and " $\succeq$ " and " $\succeq$ ". Mukund and Srihari [2009] give a more detailed explanation of the heuristic rules that are applied at the post processing stage in Section 3.3.

To incorporate replacement of the special symbols \$ and ^, with "y"/"ya" and "h" or prune these symbols from the actual transliterated word, bigram transition scores of the actual English transliterations (with the CV\* splits) were calculated using the CMU language toolkit.<sup>24</sup>

 $<sup>^{23}\</sup>mathrm{See}$  http://www.fjoch.com/GIZA++.html.

<sup>&</sup>lt;sup>24</sup>See http://svr-www.eng.cam.ac.uk/~prc14/.

Table XVIII. Similarity Scores

Similarity between	Test Set 1	Test Set 2
actual and approximate transliterated string	56.03	55.41
actual and first alignment string	64.03	63.04
actual and final transliterated string	69.18	68.92

Table XIX. Precision with Topmost Alignment Strings

Tests	Accuracy without	Found words	Accuracy after
	post-processing		post-processing
Set1	25% (highest probability)	84%	40%
	60% (top four probabilities)		
Set2	14% (highest probability)	82%	38%
	50%(top four probabilities)		

#### 10.3 Results

We tested our proposed method using two sets of 100 unique names not present in the training corpus. Neither of these sets share names between them. So our accuracies can be considered as being reported over unknown words (not seen in the training data). The similarity scores in a way determine how many of the vowels were successfully restored. Levenshtein distance between the two strings is measured in each case. The percentage similarity is computed using Equation1.

$$Similarity(\%) = \left(1 - \frac{\text{levenshtien}(str_1, str_2)}{\max(\text{strlen}(str_1), \text{strlen}(str_2))}\right) * 100$$
 (1)

where  $str_1$  and  $str_2$  are the two strings and strlen() is the length of the string. This percentage similarity is averaged over the 100 words of each test set and three cases are considered:

- (1) similarity between the required transliterated string and the approximate transliterated string,
- (2) similarity between the required transliterated string and the first alignment string without the post processing step, and
- (3) similarity between the required transliterated string and the final found string after the post processing step is applied.

The scores obtained are shown in Table XVIII.

The results show a considerable improvement in the overall similarity score after the post processing step is applied, as many alignment strings with second highest probabilities (correct transliterations) are chosen to be the most likely English transliterations. Table XIX summarizes the results over the two test sets.

In order to determine the correctness of our observations and the character mappings file thus generated, a new mappings file for ambiguous characters were generated as explained before. The transliterations obtained show a dip in the accuracy over the two data sets to 20% and 9% (highest probabilities) without the post-processing stage. Comparing the output got with the two mappings (mappings based on observations and new mappings), we see

that clearly treating "o" and "a" as treated for a pronunciation system (text to speech) is not accurate for a transliteration system. Using "j" for "o" and "o" caused names such as "yanus" and "yakoob" to be transliterated to "janus" and "jakoob". Using "y" instead of "j" in the first phase and later using heuristic rules to correct based on the phonetic constraint gives better results. Similar issues were found with "o" indicating that mappings used for name transliterations have to be dealt with separately and cannot be combined with common word pronunciations.

The uniqueness of our method is based on how the training data is prepared and alignment scores are calculated. The results reported in Tables XVIII and XIX were determined over test sets that contained only Islamic names. It was observed that this method performs poorly for Hindu names. The poor performance can be attributed to the following factors.

- (1) Our training data has no examples of Hindu names.
- (2) Many Hindu names have consonant groupings that are rarely found in common Islamic names. Consonant groups like "rv", "mr", "dr", "nk" commonly occur in Hindu names but rarely occur in Islamic names.
- (3) Most common Hindi names do not end with aspirated consonants or "h". This generates a distance measure of 1 for names like "dara".

We also tested our method on a testset of English names. "Andria", "Mathew', "Mark" were transliterated to "Inderia", "Masyo", and "Marik". It is observed that approximate transliterations are very close to the actual ones. "andria" is approximately transliterated to "Andrya", "Mathew" to "Mathyw", and "Mark" to "Maark". We also notice that the most ambiguous cases for English words are in the interpretation of "y" and "w". "y" is usually changed to "e" or "i", and "w" to "o" or retained as "w". Our approximate transliterations can be heuristically modified to get transliterations close to the actual names.

## 10.4 IPA Standards

Along with providing transliterations that are learnt using the alignment probabilities, we also provide another useful transliteration mechanism: transliterations based on IPA standards [Hussain and Afzal 2001]. Though this is based on a lookup table, the transliteration is helpful to give a user who understands Hindi readability with Urdu. For example, "عرفات": Arafat. We provide three types of transliterations—English, UrduIPA, and UrduTransliterate—for a word that is marked a proper noun. English form is obtained by performing lexicon lookups. UrduIPA form is obtained by applying IPA standards on the Urdu word, and UrduTransliterate form is obtained by applying the technique explained in Section 10. Table XX shows the different forms obtained for the example word "Arafat".

# 11. CONCLUSION

Putting together a comprehensive text analysis system for a language like Urdu is a challenging task. In this work, we have developed an NLP system for Urdu that extracts basic language related information. The tagsets used

Table XX. Transliterations Types

Normal Form type	Value	Retrieval Type
English	Arafat	Lexicon lookup
UrduIPA	[r][f][a][t]	IPA standards
UrduTransliterate	Arafaat	transliteration learning and heuristic rules

for POS, NE, and shallow parser modules are standardized and our results can now be used as baseline for further improvements. The training data that we use is freely available for comparison. We have also shown that although Urdu has limited resources (required to develop statistical models), significant information and data can be successfully borrowed from a syntactically similar language like Hindi. Experiments performed in Section 6 and 7 show that developing an Urdu specific NLP system is a must and tools developed for Hindi cannot be directly used on Urdu despite the similarity between the two languages. Our framework can be used to preprocess text and perform further analysis such as emotion detection, agent-target identification, and question opinion mining. We also provide different transliteration methods based on pronunciation and IPA standards that can improve crosslingual search and machine translation.

## **REFERENCES**

ACE. 2005. Project specifications. ACE data overview.

ADAFRE, S. F. AND RIJKE, M. 2006. Finding similar sentences across multiple languages in Wikipedia. In *Proceedings of the Conference of the European Chapter of ACL Workshop on New Text-Wikis and Blogs and Other Dynamic Text Sources (ACL'06)*.

AGI, Ž., DOVEDAN, Z., AND TADI, M. 2008. Improving part-of-speech tagging accuracy for Croatian by morphological analysis. *Informatica* 32, 445–451.

AHMAD, Z., ORAKZAI, J. K., SHAMSHER, I., AND ADNAN, A. 2007. Urdu nastaleeq character recognition. In *Proceedings of World Academy of Science, Engineering and Technology (WASET'07)*. Vol. 26.

AHMED, T. 2009. Roman to Urdu transliteration using word list. In *Proceedings of the Conference of Language and Technology (CLT'09)*.

AHN, D., JIJKOUN, V., MISHNE, G., MULLER, K., RIJKE, M., AND SCHLOBACH, S. 2004. Using Wikipedia at the TREC QA track. In *Proceedings of the Text Retrieval Conference (TREC'04)*.

AL-ONAIZAN, Y. AND KNIGHT, K. 2002. Machine transliteration of names in Arabic text. In Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages (CASL'02).

ALI, A. R. AND IJAZ, M. 2009. English to Urdu transliteration system. In *Proceedings of the Conference of Language and Technology (CLT'09)*.

AROONMANAKUL, W. 2002. Collocation and Thai word segmentation. In Proceedings of International Workshop on Spanish Language Processing and Language Technologies and International Committee for the Co-ordination and Standardisation of Speech Databases and Assessment Techniques (SNLP-COCOSDA'07).

AVINESH, P. AND KARTHIK, G. 2007. Part of speech tagging and chunking using conditional random fields and transformation-based learning. In *Proceedings of the Workshop on Shallow Parsing for South Asian Languages (IJCAI'07)*. 21–24.

AWASTHI, P., RAO, D., AND RAVINDRAN, B. 2006. Part of speech tagging and chunking with HMM and CRF. In *Proceedings of the NLPAI Machine Learning Competition*.

Babych, B. and Hartley, A. 2003. Improving machine translation quality with automatic named entity recognition. In *Proceedings of Workshop on Machine Translation and other Language Technology Tools (EAMT/EACL'03)*. 1–8.

- BAIDSY, F., HIRSCHBERG, J., AND FILATOVA, E. 2008. An unsupervised approach to biography production using Wikipedia. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL'08)*.
- Bharati, A., Sangal, R., Chaitanya, V., Kulkarni, A., Sharma, D. M., and Ramakrishna-Macharyulu, K. V. 2002. AnnCorra: building tree-banks in Indian languages. In *Proceedings* of the 3rd Workshop on Asian Language Resources and International Standardization of the Association of Computational Linguistics (ALR'02). 12, 1–8.
- BHARATI, A., SHARMA, D. M., BAI, L., SANGAL, R., AND IIIT, H. 2006. AnnCorra: Annotating corpora guidelines for POS and chunk annotation for Indian language. Tech. rep. Language Technologies Research Centre, IIIT, Hyderabad.
- 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 3rd Linguistic Annotation Workshop (The LAW III) in conjunction with the Association for Computational Linguistics/International Joint Conference on Natural Language Processing (ACL/IJCNLP'09)*.
- BIKEL, D. M., SCHWARTZ, R. L., AND WEISCHEDEL, R. M. 1999. An Algorithm that Learns What's in a Name. *Mach. Learn.* 34, 1–3, 211–231.
- BÖGEL, T., BUTT, M., HAUTLI, A., AND SULGER, S. 2007. Developing finite state morphological analyzer for Hindi and Urdu. In *Proceedings of the Finite-State Methods and Natural Language Processing 6th International Workshop (FSMNLP'07)*.
- BORTHWICK, A. 1999. Maximum entropy approach to named entity recognition. PhD thesis, New York University.
- Brants, T. 2000. TNT A statistical part-of-speech tagger. In *Proceedings of the 6th Applied Natural Language Processing Conference (ANLP'00)*.
- Brill, E. 1992. A simple rule-based part of speech tagger. In *Proceedings of the Workshop on Speech and Natural Language Processing (SNLP'92)*. 116.
- BUSCALDI, D. AND ROSSO, P. 2006. Mining knowledge from Wikipedia for the question answering task. In *Proceedings of the 5th International Conference on Language Resources and Evaluation* (LREC'06).
- Butt, M. and King, T. H. 2002. Urdu and the Parallel Grammar Project. In *Proceedings* of the COLING Workshop on Asian Language Resources and International Standardization (COLING'02). 39–45.
- BUTT, M. AND KING, T. H. 2004. The status of case. In V. Dayal, and A. Mahajan, Eds. *Clause Structure in South Asian Languages*, Studies in Natural Language and Linguistic Theory Series, vol. 61.
- CENTER FOR RESEARCH IN URDU LANGUAGE PROCESSING (CRULP). 2007. Urdu Component Development Project, Urdu Normalization Application v1.0.
- Choi, K. S., Isahara, H., Kanzaki, K., Kim, H., Pak, S. M., and Sun, M. 2009. Word segmentation standard in Chinese, Japanese, and Korean. In *Proceedings of the 7th Workshop on Asian Language Resources (ACL-IJCNLP'09)*. 179–186.
- Daelemans, W., Buchholz, S., and Veenstra, J. 1999. Memory-based shallow parsing. In *Proceedings of the Conference on Natural Language Learning (CoNLL'99)*. 53–60.
- DANDAPAT, S., SARKAR, S., AND BASU, A. 2007. Automatic part-of-speech tagging for Bengali: An approach for morphologically rich languages in a poor resource scenario. In *Proceedings of the Association of Computational Linguistics (ACL'07)*. 221–224.
- DURRANI, N. 2007. Typology of word and automatic word segmentation in Urdu text corpus. National University of Computer and Emerging Sciences, Lahore, Pakistan.
- Durrani, N. and Hussain, S. 2010. Urdu word segmentation. In *Proceedings of the 11th Annual Conference of Human Language Technology Conference North American Chapter of the Association for Computational Linguistics (HLT-NAACL'10)*.
- EKBAL, A. AND BANDYOPADHYAY, S. 2007. A hidden Markov model based named entity recognition system: Bengali and Hindi as case studies. Lecture Notes in Computer Science vol. 4815, 545.
- ACM Transactions on Asian Language Information Processing, Vol. 9, No. 4, Article 15, Pub. date: December 2010.

- EKBAL, A. AND BANDYOPADHYAY, S. 2009. A conditional random field approach for named entity recognition in Bengali and Hindi. Linguist. Issues Lang. Technol. 2, 1.
- EKBAL, A., AND BANDYOPADHYAY, S. 2010. Named entity recognition using support vector machine: A language independent approach. Int. J. Elec. Comput. Syst. Eng. 4.
- ETHNOLOGUE: LANGUAGES OF THE WORLD, 14TH Ed. 2002. http://www.ethnologue.com/show\_country.asp?name=Pakistan (accessed 3/03).
- ERIK F., SANG T. K., AND DÉJEAN, H. 2001. Introduction to the CoNLL Shared Task: Clause Identification. In Proceedings of the Conference on Natural Language Learning (CoNLL'99). 53 - 57
- FARMAKIOTOU, D., KARKALETSIS, V., KOUTSIAS, J., SIGLETOS, G., SPYROPOULOS, C. D., AND STAMATOPOULOS, P. 2000. Rule-based named entity recognition for Greek financial texts. In Proceedings of the Workshop on Computational Lexicography and Multimedia Dictionaries (COMLEX'00). 75-78.
- GAO, J. F., LI, M., WU, A., HUANG, C. N. 2005. Chinese word segmentation and named entity recognition: A pragmatic approach. Comput. Linguist.
- GARRIDO-ALENDA, A., GILABERT-ZARCO, P., PÉREZ-ORTIZ, J. A., PERTUSA-IBÁÑEZ, A., RAM REZ-SANCHEZ, G., SANCHEZ-MARTINEZ, F., SCALCO, M. A., AND FORCADA, M. L. 2004. Shallow parsing for Portuguese-Spanish machine translation. In A. Branco, A. Mendes, and R. Ribeiro Eds., Language Technology for Portuguese: Shallow Processing Tools and Resources, 135 - 144.
- HAN C., HAN, N. R., KO, E. S., AND PALMER, M. 2002. Development and evaluation of a Korean treebank and its application to NLP. Lang. Inform. 6, 1, 123-138.
- HAQ, A. M. 1987. اردو صرف و نحو. Amjuman-e-Taraggi Urdu (Hindi).
- HARDIE, A. 2003. Developing a tagset for automated part-of-speech tagging in Urdu. Department of Linguistics and Modern English Language, University of Lancaster.
- HUMAYOUN, M. 2006. Urdu morphology, orthography and lexicon extraction. Master's Thesis. Department of Computing Science, Chalmers University of Technology.
- HUMAYOUN, M., HAMMARSTRÖM, H., AND RANTA, A. 2007. Urdu morphology, orthography and lexicon extraction. In Proceedings of the 2nd Workshop on Computational Approaches to Arabic Script-based Languages (CAASL'07).
- HUSSAIN, S. 2004. Letter-to-sound conversion for Urdu text-to-speech system. In Proceedings of Workshop on Computational Approaches to Arabic Script-based Languages (COLING'04). 74-49.
- HUSSAIN, S. 2008. Resources for Urdu language processing. In Proceedings of the 6th Workshop on Asian Language Resources (IJCNLP'08).
- HUSSAIN AND AFZAL, M. 2001. Urdu computing standards: Urdu Zabta Takhti (UZT 1.01). In  $Proceedings\ of\ IEEE\ International\ Multi-Topic\ Conference\ (INMIC'01).$
- JANUS, L. 1997. Less commonly taught languages of emerging importance: Major issues, cost problems, and their national implications. In Proceedings of a National Policy Conference on Higher Education Act, Title VI and Fullbright-Hays Program.
- JAVED, I. 1981. Taraqqi Urdu Bureau, New Delhi.
- JIANG, J. AND CONRATH, W. 1997. Semantic similarity based on corpus statistics and lexical taxonomy.
- KANAAN, G., AL-SHALABI, R., AND SAWALHA, M. 2005. Improving Arabic information retrieval systems using part of speech tagging. Inform. Technol. J. 4. 32-37.
- KARIMPOUR, R., GHORBANI, A., PISHDAD, A., MOHTARAMI, M., ALEAHMAD, A., AMIRI, H., AND OROUMCHIAN, F. 2008. Using part of speech tagging in Persian information retrieval. CLEF
- KASHANI, M., POPOWICH, F., AND SADAT, F. 2007. Automatic transliteration of proper nouns from Arabic to English. The Challenge of Arabic for NLP/MT, 76-84.
- KING, C. K. 1995. One language, two scripts: The Hindi movement in 19th century North India. Annual Urdu Studies 10.
- KO, J., MITAMURA, T., AND NYBERG, E. 2007. Language-independent probabilistic answer ranking for multilingual question answering. In Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL'07).
- ACM Transactions on Asian Language Information Processing, Vol. 9, No. 4, Article 15, Pub. date: December 2010.

- LAFFERTY, J. D., McCallum, A., Pereira, C. N. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning (ML'01)*. 282–289.
- LEECH, G. AND WILSON, A. 1999. Standards for tagsets. Edited version of EAGLES recommendations for the morphosyntactic annotation of corpora. In van Halteren, H. Ed., *Syntactic Wordclass Tagging*, Kluwer Academic Publishers, Dordrecht.
- Liao, P. P., Liu, Y., and Chen, L. 2006. Hybrid Chinese text chunking. In *Proceedings of the IEEE International Conference on Information Reuse and Integration (CIRI'02)*. 561–566.
- MAAMOURI, M., BIES, A., BUCKWALTER, T., AND MEKKI, M. 2004. The Penn arabic treebank: Building a large-scale annotated arabic corpus. In *Proceedings of the NEMLAR Conference on Arabic Language Resources and Tools (ALRT'04)*. 102–109.
- MARCUS, M. P., SANTORINI, B., AND MARCINKIEWICZ, M. A. 1994. Building a large annotated corpus of English: The Penn treebank. *Comput. Linguist.* 19, 313–330.
- McCallum, A. and Li, W. 2003. Early results for named entity recognition with conditional random fields, feature induction and Web-enhanced lexicons. In *Proceedings of the Conference on Natural Language Learning (CoNLU3)*.
- MEKNAVIN, S., CHAROENPORNSAWAT, P., AND KIJSIRIKUL, B. 1997. Feature-based Thai word segmentation. In *Proceedings of the Natural Language Processing Pacific Rim Symposium* (NLP'97).
- MISHRA, N., YADAV, S., AND SIDDIQUI, T. J. 2009. An unsupervised approach to Hindi word sense disambiguation. In *Proceedings of the 1st International Conference on Intelligent Human Computer Interaction (IHCl'09)*. 327–335.
- MITRA, M. AND MAJUMDER, P. 2008. Forum for information retrieval evaluation (for Indian languages). Cross Language Evaluation Forum. Indian Statistical Institute, Calcutta.
- MOLDOVAN, D., HARABAGIU, S., GIRJU, R., MORARESCU, P., LACATUSU, F., NOVISCHI, A., BADULESCU, A., BOLOHAN, O. 2002. LCC tools for question answering. In *Proceedings of the Conference on Text REtrieval (TREC'02)*.
- Muaz, A., Ali, A., and Hussain, S. 2009. Analysis and development of Urdu POS tagged corpus. In *Proceedings of the 7th Workshop on Asian Language Resources (ACL-IJCNLP'09)*. 24–31.
- Muaz, A. and Khan, A. N. The morphosyntactic behavior of "Wala" in Urdu language.
- MUKUND, S., PETERSON, E., AND SRIHARI, R. K. 2009. Context aware transliterations of Urdu names. In *Proceedings of the 7th International Conference on Natural Language Processing (ICNLP'09)*.
- MUKUND, S. AND SRIHARI, R. K. 2009. NE tagging for Urdu based on bootstrap POS learning. In Proceedings of the 3rd International Workshop on Cross Lingual Information Access: Addressing the Information Need of Multilingual Societies (CLIAWS3'09).
- NICHOLLS, C., AND SONG, F. 2009. Improving sentiment analysis with part-of-speech weighting. In *Proceedings of the 8th International Conference on Machine Learning and Cybernetics (ICMLC'09)*.
- PANDIAN, S. L. AND GEETHA, T. V. 2009. Semantic role labeling for Tamil documents. *Int. J. Recent Trends Eng. 1*, 1.
- PAPAGEORGIOU, C. 1994. Japanese Word segmentation by hidden Markov model. In *Proceedings* of the Human Language Technologies Workshop (HLT'94). 283–288.
- PLATTS, T. 1967. A Grammar of the Hindustani or Urdu Language. Munshiram Manoharlal Delhi.
- PORNPRASERTKUL, A. 1994. Thai syntactic analysis. Ph.D Thesis, Asian Institute of Technology.
- Punyakanok, V., Roth, D., and Yih, W. T. 2005. The necessity of syntactic parsing for semantic role labeling. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAP05)*. 1117.
- RAMSHAW, L. A. AND MARCUS, M. P. 1995. Text chunking using transformation-based learning. In Proceedings of the 3rd Workshop on Very Large Corpora (VLC'95). 82–94.
- RATNAPARKHI, A. 1996. A maximum entropy model for part-of-speech tagger. In *Proceedings of the Empirical Methods in Natural Language Processing Conference (EMNLP'96)*.
- ACM Transactions on Asian Language Information Processing, Vol. 9, No. 4, Article 15, Pub. date: December 2010.

- RICHMAN, A. AND SCHONE, P. 2008. Mining Wiki resources for multilingual named entity recognition. In Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL'08).
- RIZVI, J. S., HUSSSAIN, M., AND QAISER, N. 2004. Language oriented parsing through morphologically closed word classes in Urdu. In Proceedings of the Engineering, Sciences and Technology Student Conference (SCONEST'04). 19-24.
- RUSSELL, R. 1996. Some notes on Hindi and Urdu. Annual Urdu Studies 11, 203-8.
- SAHA, S. K., MITRA, P., AND SARKAR, S. 2008. Word clustering and word selection based feature reduction for MaxEnt based Hindi NER. In Proceedings of the Conference on Human Language Translations (HLT'08). 488-95.
- SAJJAD, H. 2007. Statistical part of speech tagger for Urdu. Unpublished Master's Thesis. National University of Computer and Emerging Sciences. Lahore, Pakistan.
- SAJJAD, H. AND SCHMID, H. 2009. Tagging Urdu text with parts of speech: A tagger comparison. In Proceedings of the 12th Conference of the European Chapter of the Association for Computation Linguistics (EACL'09). 692–700.
- SAMY, D., MORENO, A., AND GUIRAO, M. 2005. A proposal for an Arabic named entity tagger leveraging a parallel corpus. In Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP'05).
- SANTORINI, B. 1990. Part-of-Speech Tagging Guidelines for the Penn Treebank Project 3rd Ed. University of Pennsylvania.
- SCHMID, H. 1994. Probabilistic Part-of-Speech Tagging Using Decision Trees. Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart, Germany.
- SCHMIDT, R. 1999. Urdu: An Essential Grammar. Routledge, London.
- SHAH, C. AND BHATTACHARYYA, P. 2002. A study for evaluating the importance of various parts of speech (POS) for information retrieval (IR). In Proceedings of the International Conference on Universal Knowledge and Language (ICUKL'02).
- SHAMSFARD, M. AND MOUSAVI, M. S. 2008. Thematic role extraction using shallow parsing. Int. J. Computat. Intell.
- SIDDIQI, A. 1971. Jaame-ul-Qwaid Head of the Department of Urdu, Karachi University, Markazi Urdu Board, Lahore.
- SINGH, A. K. 2006. Study of some distance measures for language and encoding identification. In Proceedings of the ACL Workshop on Linguistic Distances (LD'06).
- SINGH, A. K. AND SURANA, H. 2007. Using a single framework for computational modeling of linguistic similarity for solving many NLP problems. In Proceedings of Eurolan Doctoral Consortium (EUROLAN'07).
- SINHA, R. M. K. 2009. Mining complex predicates in Hindi using a parallel Hindi-English corpus. In Proceedings of the ACL International Joint Conference in Natural Language Processing (LICNLP'09), 40.
- SPROAT, R. AND SHIH, C. 2002. Corpus-based methods in Chinese morphology and phonology. In Proceedings of the 19th International Conference on Computational Linguistics (COOLING).
- SRIHARI, R. K., LI, W., NIU, C., AND CORNELL, T. 2008. InfoXtract: A customizable intermediate level information extraction engine. J. Nat. Lang. Eng. 14, 1, 33-69.
- STALLS, B. G. AND KNIGHT, K. 1998. Translation name and technical terms in Arabic text. In Proceedings of the COLING/ACL Workshop on Computational Approaches to Semitic Languages (CAASL'98).
- STANISLAV, M. 2003. Statistical approach to the debate on Urdu and Hindi (student paper). Annual Urdu Studies 18.
- STYMNE, S., HOLMQVIST, M., AND AHRENBERG, L. 2008. Effects of morphological analysis in translation between German and English. In Proceedings of the 3rd Workshop on Statistical Machine Translation (SMT'08), 135–138.
- URDU COMPONENT DEVELOPMENT PROJECT, APPLICATION PROGRAMMING INTERFACE 2007. Normalization Utility, Center for Research in Urdu Language Processing, National University of Computer and Emerging Sciences, Lahore, Pakistan.
- ACM Transactions on Asian Language Information Processing, Vol. 9, No. 4, Article 15, Pub. date: December 2010.

- WANG, X. J., QIN, Y., AND LIU, W. 2007. A search-based Chinese word segmentation method. In Proceedings of the 16th International Conference on the World Wide Web (WWW'07). 1130.
- WILKS, Y. AND STEVENSON, M. 1996. The grammar of sense: Is word-sense tagging much more than part-of-speech tagging?
- XUE, N. W. 2003. Chinese word segmentation as character tagging. Comput. Linguist. Chinese Lang. Proc. 8, 1, 29–48.
- XUE, N., XIA, F., CHIOU, F. D., AND PALMER, M. 2005. The Penn Chinese treebank: Phrase structure annotation of a large corpus. *Nat. Lang. Eng.* 11, 207–238.
- ZHOU, G. AND SU, W. 2002. Named entity recognition using an HMM-based chunk tagger. In *Proceedings of the 40th Annual Meeting on Association For Computational Linguistics*. Association for Computational Linguistics, 473–480.

Received September 2009; revised March 2010; accepted April 2010