Developing Language Technology Tools and Resources for Sindhi: A Resource-Poor Language

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
in
Computational Linguistics by Research

by

Raveesh Motlani
201125035
raveesh.motlani@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
July 2018

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled " **Developing Language Technology Tools and Resources for Sindhi: A Resource-Poor Language** " by **Raveesh Motlani**, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Adviser: Prof. Dipti Misra Sharma

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled " **Developing Language Technology Tools and Resources for Sindhi: A Resource-Poor Language** " by **Raveesh Motlani**, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Co-Adviser: Dr. Manish Shrivastava

To all my friends and family.

Acknowledgments

I would like to express my deep sense of gratitude to my advisor Prof. Dipti Misra Sharma for letting me be a part of the CLD batch of 2011. The initial years in the college were academically difficult for me. I am grateful to you for believing in me during those tough times.

Thank you, Dr. Manish Shrivastava, for being a friend and a guide. Our discussions have helped me and my work grow a lot and become what it is today. You have been around whenever I wanted someone to discuss the sparking ideas in my head. Thank you for being there at odd times, encouraging my thoughts and helping shape them into better ideas.

This research work would not have reached where it did without the help and support from various people in the Sindhi community. I would like to thank Pinkey Nainwani for collaborating with me in creation and annotation of Sindhi Devanagari data. I would also like to thank Mehtab Ahmed Solangi for helping me with annotation and translation of several words in the Sindhi language. I am also indebted to Mr. Chunnilal Wadhwani (for teaching me reading and writing in Perso-Arabic script), Mr. Bhagwan Babani and Mr. Gyanchand Lalwani for helping me collected raw data in Sindhi.

I met really interesting people and made a lot of friends during the my journey at IIIT-Hyderabad. These friends have been an important my life by being there to morally support me in tough times, motivate me to achieve more and helping increase my capabilities. I would like acknowledge these amazing people I met at IIIT-H, specially Juhi Tandon, Arnav Sharma, Nehal J Wani, Himanshu Sharma, Diksha Yadav and Dr. Francis Tyers, who helped me in improving my research, proofread my work, gave valuable feedback and spend their time discussing various ideas. Talking about friends and amazing people, I would like to mention Nikhar, Vibhav, Anurag, Shiv, Akash, Kumar, Tiwari, Sakshi, Vishrut, Akshat, Prateek and all my batchmates of UG2k11 for being there. Thank you so much guys.

Last but not the least, I am thankful to family for their support, without them I could not have gathered the courage and take the opportunity to go against the traditions and pursue what I wanted to.

Abstract

Sindhi is an Indo-Aryan language, which is spoken by about 53 million people in Pakistan and about 5.8 million people in India. Sindhi is also one of the 22 official languages in India. Despite all these statistics showing how widely spoken Sindhi is, it is still a computationally resource poor language.

Development of natural language applications for any language is possible with the help of linguistic resources and computational tools for that language. In this work, we have developed some fundamental resources and tools that shall help natural language processing of Sindhi language. We have developed raw and part-of-speech (POS) annotated corpus for Sindhi Devanagari and subsequently created a Conditional Random Fields (CRF) based automatic POS Tagger that yields an accuracy of 91.78%.

We have also built a paradigm based finite-state morphological analyser for Sindhi Perso-Arabic using Apertium's lttoolbox. This morphological analyser currently has about 3500 entries and a coverage of more than 81% on Sindhi Wikipedia consisting of 341.5k tokens. We worked on Sindhi Perso-Arabic because the corpus of Sindhi Devanagari was very small and we needed very large corpus for good coverage of the vocabulary of the language, which was available in Perso-Arabic.

To diminish the script barrier, we also worked on transliteration. We developed a rule-based transliteration system between Sindhi Devanagari and Sindhi Perso-Arabic which yields 91.33% accuracy. We have also conducted experiments to demonstrate resources leveraging and sharing among the scripts through transliteration. These include, generating more data for Sindhi Devanagari to bootsrap POS tagger and building POS tagger for Sindhi Perso-Arabic.

Contents

List of Figures

List of Tables

Chapter *1*

Introduction

Language technology tools and resources are vital assets that ensure digital existence of a language for a long time. These tools facilitate the understanding of natural languages by computers. Such tools and resources are necessary for natural language processing and have aplenty applications in the digital era. For instance, cross-lingual technologies such as machine translation help people across the world communicate with each other using their native languages and access information present in a language they do not know. Similarly, automatic speech recognition helps people communicate with machines by speaking in their natural languages. There can be many more such applications where a better understanding of natural languages by machines could be helpful in communicating and sharing knowledge across billions of people in the world and diminishing the language barrier.

A lot of popular languages in the world are equipped with resources and tools to develop such applications and facilitate further research on more challenging problems in computational linguistics but a larger set of languages in this world are devoid of even fundamental resources required for developing tools and applications in the field of natural language processing. Therefore, it is important to protect such languages from being digitally endangered.

Our work is based on one such resource-poor language, Sindhi. Our aim is to develop some basic resources and tools which shall facilitate natural language processing for Sindhi and help in extending its digital existence.

1.1 The Sindhi Language

Sindhi is an Indo-Aryan language, which is spoken by about 53 million people in Pakistan and about 5.8 million people in India. Sindhi is also one of the 22 official languages in India[1]. Despite all these statistics showing how widely spoken Sindhi is, it is still a computationally resource poor language.

Historically, Sindhi has been written using many writing systems such as Landa, Waranki, Khudawadi, Gurmukhi, Perso-Arabic and Devanagari. Currently, only Devanagari and Perso-Arabic are the only prevalent scripts for writing texts in Sindhi. The Sindhi alphabet in Devanagari shares almost all the letters of Hindi alphabet and also has additional 4 letters for Sindhi implosives. Similarly, the Sindhi alphabet in Perso-Arabic is a variant of the Persian alphabet and shares many characters with Arabic and Persian alphabets. It has eighteen other letters (see Table 6.1) to capture the sounds particular to Sindhi (implosives, retroflex and nasal sounds).

The literature [13] says that during the colonial rule, the British regime faced a problem in recognizing the major prevalent script out of many others for Sindhi and after prolonged deliberation they chose Perso-Arabic as the official script for Sindhi in 1853.

Later, when the partition of India and Pakistan took place in 1947, Sindh (the region where majority of Sindhi speaking people resided) became a part of Pakistan. A lot of Sindhi speaking people migrated to India and spread across the country. When the question of declaring a standard script for Sindhi in India came up, groups supporting either Perso-Arabic or Devanagari stood up. Initially, the Indian Government declared Devanagari as the standard script of Sindhi but owing to protests, both scripts were eventually accepted. However, the Perso-Arabic script remained standard in Pakistan.

The Sindhi speaking population do not have a geographical state in India. Hence, despite being a scheduled language in the Indian Constitution, it is not used as an official language anywhere in the country and therefore, does not have a huge literature. On the other hand, there is Sind, an official state in Pakistan and people have been contributing to the literature of the language in both printed and digital forms. For instance, there exists a Sindhi Wikipedia, various news websites [2] and blogs in Perso-Arabic Sindhi, that are published from Pakistan. In contrast, there is very little digitally accessible text of Sindhi in Devanagari (henceforth Sindhi-Devanagari) on the web.

[1] The Constitution of India. page 330, EIGHTH SCHEDULE, Articles 344 (1) and 351. Languages.
[2] http://www.abyznewslinks.com/pakis.htm

1.2 Low-Resource NLP: Importance and Challenges

We are living in a digital era where technology is growing exponentially and alleviating problems faced by humans increasingly. Natural Language Processing (NLP) aims to bridge the gap between technology and human (or natural) languages by enabling computers to understand them. It also helps to convert natural language inputs into computational instructions. The natural language interfaces also make interaction with computers more accessible and easy for humans. NLP also empowers the languages - it can now survive longer, thanks to the digital technology.

NLP is a hard problem which requires enormous amount of digital data as well as linguistic information about a natural language to comprehend it at phonetic, syntactic, semantic, pragmatic and other levels. The absence of digital data (audio, video or text), sufficient corpora (parallel, comparable, annotated or raw machine readable data) and linguistic resources (dictionaries, grammar, etc) impedes creation of NLP applications.

This, in turn, poses a big challenge to NLP for languages that do not have enough digital data or linguistic resources. These languages are referred to as resource-poor languages. Currently, there are about 7,000 languages in the world out of which approximately 40 languages are considered resource-rich. Clearly, it is a matter of concern for rest of the languages. Many languages across the globe are dying rapidly. It has been predicted that half of the languages shall be extinct by the end of this century.

Thus, creation of linguistic resources and NLP tool is crucial for developing natural language applications and help them survive.

1.3 Problem Statement

In the previous sections, we have discussed why creation of language technology tools is important and how immensely it contributes to the survival of a language in the rapidly growing digital world. Apart from that, it will also enable communication with computers and other humans without language barriers.

The goal of this research is to equip Sindhi with certain linguistic tools, resources and data. The research holds significance as Sindhi is a resource-poor language. The creation of these fundamental resources shall facilitate NLP in Sindhi and enable development of natural language applications.

1.4 Summary of Major Contributions

Our research was aimed at developing fundamental language processing tools and resources for Sindhi, which is a resource-poor language. The following are some major contributions of our research:

1. **Part-of-Speech Tagger** : We have developed an annotated corpus and a statistical POS Tagger for Sindhi-Devanagari using Conditional Random Fields, which yeilds an average accuracy of 91.78% (10-fold cross-validation).

2. **Morphological Analyser** : We used Apertium's lttoolbox [21] to develop a paradigm based finite-state morphological analyser for Sindhi Perso-Arabic. This morphological analyser currently has about 3500 entries and a coverage of more than 81% on Sindhi Wikipedia consisting of 341.5k tokens.

3. **Transliteration System** : We developed a transliteration system to convert texts in Sindhi Devanagari to Sindhi Perso-Arabic and diminish the script barrier. The system is developed through rule-based technique and transliterates with an accuracy of 91.33%.

1.5 Organization of the Thesis

This thesis is divided into 5 chapters. Here is what we have discussed in each of them.

- **Chapter 2 - Sindhi: An Overview**: In this chapter, we have given an overview of Sindhi. We have described the background and history of the Sindhi community, their geographical origin and their language. We have also discussed some of the key incidents observed in the evolution of Sindhi language.

- **Chapter 3 - Related Work**: In this chapter, we have briefly discussed the major approaches and their related literature for building part-of-speech tagger, morphological analyzer and transliteration system. We have also discussed in detail the relevant literature for Sindhi and described other NLP tools that people have developed for Sindhi.

- **Chapter 4 - Part of Speech Tagging for Sindhi**: In this chapter, we have described the procedure for developing a POS Tagger from scratch. We have mentioned our process of raw corpus construction, followed by annotation and then experimenting with various features to create a POS Tagger using Conditional Random Fields.

- **Chapter 5 - Morphological Analyzer for Sindhi**: We have briefly described Sindhi's Morphology in the beginning of this chapter and later discussed how we constructed a paradigm based finite-state morph analyzer for Sindhi using Apertium.
- **Chapter 6 - Transliteration**: In this chapter we have described our work on developing a rule-based transliteration system to convert text from Sindhi Devanagari to Sindhi Perso-Arabic. We have discussed the various challenges with the scripts of Sindhi and how we handled some of them. We have also conducted experiments to show how transliteration can be used to leverage resources in either scripts by breaking the script barrier. Some of these experiments include building a POS Tagger for Sindhi Perso-Arabic and bootstraping more data for Sindhi Devanagari data from Perso-Arabic texts.
- **Chapter 7 - Conclusion and Future Directions**: Here, we have concluded upon the work done in this thesis and discussed future directions of research in NLP for Sindhi using our work.

Chapter *2*

Sindhi: An Overview

Sindhi language and culture was originated from the Sindh region which lies in the North-Western area of the Indian subcontinent. In this chapter, we shall discuss about the language, culture, geographical region. We shall also describe the challenges faced by Sindhi community and language which were the result of some crucial historical developments. The linguistic properties of Sindhi shall be discussed in the subsequent chapters.

2.1 Sindh Region and Sindhi People

Historically, Sindh is referred to the region situated along the North-Western border of the Indian subcontinent. Presently, it is referred to a province in Pakistan. The recorded history of Sindh region can be traced back to Mahabharata (the Indian mythological epic) era and Indus-Valley Civilization (2500 BC). Theories have been proposed that this when the Sindhi people and culture came into existence. Mohenjo-Daro (which means 'mound of the dead' in Sindhi) is considered the antiquity of Sindhi culture. However, substantial facts or proofs to draw conclusive links of Sindhi culture, people or language to the Indus Civilization are not available.

Sindh was administered by Islamic rulers since 711 AD for around 1100 years until the British conquest. Thus, Sindhi culture and language has been immensely influenced by Islamic culture and beliefs. Nevertheless, Hindu Sindhis did exist in minority in the region. Sindh was conquered by the Britishers in 1843 . In 1936, it was declared as a separate province and was given its own Assembly.

In 1940s, the Indian freedom struggle started to gain huge momentum. Hindu-Muslim relations also faced a rift as the Muslim elite started posing threats to Hindu majority in India and proposed a resolution (Lahore Resolution or Pakistan Resolution) demanding separate "inde-

pendent states" for Muslims. This resolution was passed in the Sindh Assembly in 1947 [36]. Subsequently, Sindh became a part of Pakistan, a new nation formed out of British India.

The partition of India and Pakistan created tremendous tension between Hindus and Muslims - it has not ceased to exist till date. Post partition, there were several riots and huge bloodshed, which caused the Hindus (including Sindhis) to flee their ancestral home and property in newly created Pakistan and move to India to start a new beginning. The Hindu Sindhis moved to various parts of the country, specially the western states of Maharashtra and Gujarat. Some of the refugee camps, where majority Sindhi population resided, have now settled and grown into towns such as Ulhasnagar and Kalyan, the towns adjacent to Mumbai. According to the 1951 Indian census, 337,000 Sindhi refugees had arrived in Western India post independence.

Presently, Sindh is one of the four provinces of Pakistan. The biggest portion of Sindhi community continues to reside in Sindh. Although the numbers are lower than before as around 800,000 Hindu Sindhis fled to India post partition. As per Pakistan Bureau of Statistics (1998) Sindh is still a home for the minority Hindu population. Their population is around 2 million and most of them speak Sindhi as their primary language.

2.2 Sindhi Language

The Sindhi language has been classified as a Indo-Aryan language (see Figure 2.1). It is considered to be descendant of a form of Prakrit. As suggested earlier, the term "Sindhi'' is derived from "Sindhu'' which is the local name of river Indus.

Sindhi is the official language of Sindh province in Pakistan. In India, it is one of the 22 scheduled languages. However, it is not an official language of any of the Indian states. It is spoken by approximately 53 million people in Pakistan and about 5.8 million people in India.

There are various dialects of Sindhi spoken in Pakistan namely Siroli, Lari, Lasi, Thari and Vicholi. The one being spoken predominantly in present day is Vicholi. Some dialects are also spoken in Indian states closer to Pakistan and Sindh's border, such as Kachchhi (in Gujarat) and Jaisalmeri (in Rajasthan).

Sindhi has undergone a lot divergence over time. Since Sindh was ruled by Muslim rulers for about 1100 years before the British invasion, a lot of Arabic and Persian words as well as phonemes were introduced into the language. Post independence, Sindhi spoken in India kept on diverging from what has been spoken in Pakistan. Their current vocabularies of Indian and Pakistani Sindhi have considerably influenced by the local languages, that is, Urdu and Hindi, respectively.

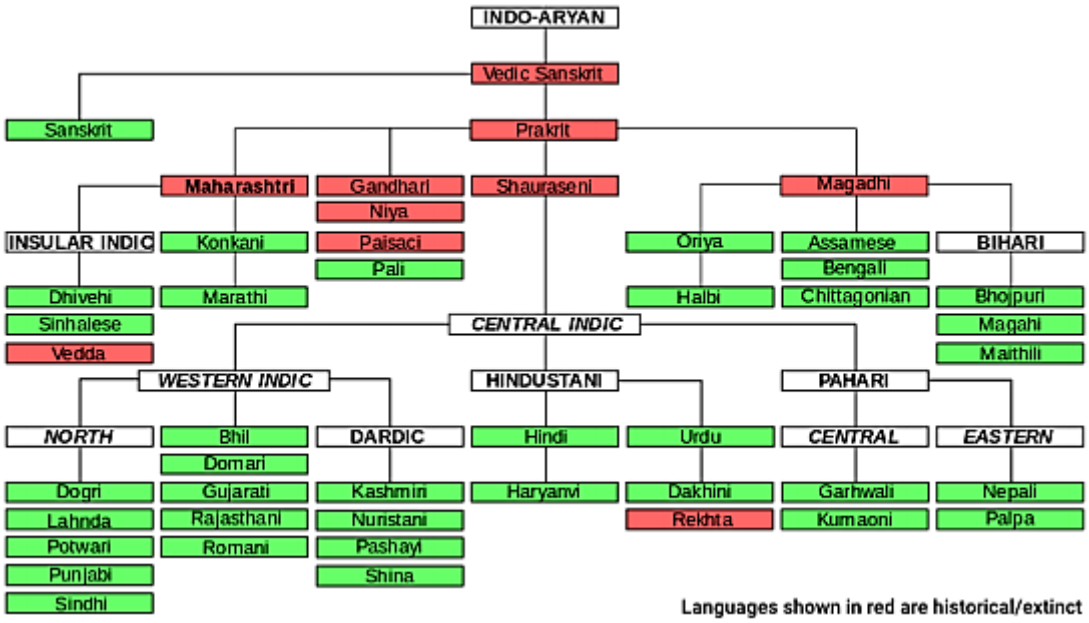


Figure 2.1: Indo-Aryan Language Family

Historically, Sindhi has been written using many writing systems such as Landa, Waranki, Khudawadi, Gurmukhi, Perso-Arabic and Devanagari. Presently, Devanagari and Perso-Arabic are the prevalent scripts for writing in Sindhi. The Sindhi alphabet in Devanagari shares almost all the letters of Hindi alphabet and also has additional 4 letters for Sindhi implosives. Similarly, the Sindhi alphabet in Perso-Arabic is a variant of the Persian alphabet and shares many characters with Arabic as well as Persian alphabets. It has eighteen other letters to capture the sounds (phonemes) which are specific to Sindhi such as implosives, retroflex and nasals.

Since Sindhi is an official language of the Sindh province, it is used abundantly in local administration, education, media and communication (both oral and written). Perso-Arabic script is the standard writing form used for Sindhi, which is also similar in appearance to Urdu, the national language of Pakistan.

As stated earlier, although Sindhi is an officially recognized language by the Central Government of India, it is not an official language of any of the states of India. Initially, the Indian Government declared Devanagari as the standard script for Sindhi, which is very similar in appearance and structure to Devanagari script. Devanagari is used to write Hindi, the predominant official language of India. Later, Perso-Arabic script was also declared as a standard in India, owing to the protests staged by proponents of Perso-Arabic script. It should be noted that Perso-Arabic was declared as the official script for Sindhi by the British Government in 1853.

Thus, Sindhi literature has flourished a lot more in Pakistan as compared to India. Sindhi community in Pakistan has also significantly contributed to Sindhi digital content creation in the form of blogs, newspaper articles and Sindhi Wikipedia. In India, however, there is hardly any digital content available in Sindhi Devanagari or Perso-Arabic script. Some possible reasons for underutilization of Sindhi text could be the script-conflict, minimal usage in education and administration.

2.3 Summary

Thus, it is clear that Sindh region and Sindhi language have had long and rich history. Sindhi has had Arabic and Persian influence on the language and script due to several invasions by Islamic rulers. Later, Sindhi community and language diverged due to partition of British India. Having multiple scripts and non-existence of Sindhi state in India adversely affected the growth of the language. Therefore, there is a need to preserve the dying language. It can be preserved through development of linguistic resources and language understanding tools.

Chapter *3*

Related Work

3.1 Part of Speech Tagging

A lot of research exists on the problem of automatic Part-of-Speech tagging. Traditionally, many different approaches have been used for part of speech tagging. One such linguistically motivated approach is that of a rule based part-of-speech tagger.

3.1.1 Rule-Based Approach

In this approach, a large set of hand-written rules are defined by language experts and linguists, based on the linguistic properties and contextual patterns found in the language. The POS tag for an input word is disambiguated based on these rules that it follows. An example of a rule from English language could be: if the preceding word is an *article*, then the current word should be a *noun*. This classical approached was explored a lot in the initial days of automatic POS tagging research, during mid-sixties and seventies [32, 40, 28]. TAGGIT [28] was one such pioneering rule-based tagger, followed by ENGTWOL [39]. The benefit of linguistic taggers is they can tag with very high accuracy. They typically require a lot of linguistic expertise to develop rules ranging from few hundred to thousands, which takes years to build. Thus, rule-based approach is very costly and language dependent.

3.1.2 Machine Learning Approach

On the other hand, there are statistical approaches to develop POS Taggers. This approach is more popular nowadays as the cost involved is less compared to rule-based approach. There are machine learning algorithms which have been used for POS Tagging, such as Decision Trees

[7], HMMs [12, 8], MEMMs, CRFs [44], Maximum Entropy [77], etc. Sometimes, people also combine both to form a Hybrid tagger, such as CLAWS [24].

The algorithm that we have used for our work is Conditional Random Fields. Conditional Random Fields (CRFs) have been used for creating taggers for a long time now. They were first used for POS tagging experiments by [44]. Then, they were also used for the task of shallow parsing by [79] where CRFs were applied for Noun Phrase (NP) chunking of English on Wall-Street Journal corpus and reported an accuracy of 94.38%. They were also used for POS tagging several Indian languages. This is because Indian languages are morphologically rich and CRFs give the freedom to incorporate it and other linguistic properties as features while training a POS tagger.

3.1.3 POS Tagging for Indic Languages

Part-of-Speech Tagging for Indian languages has recieved a lot of attention in academia within the past 20 years. Some of the earliest of works include Bharati et al. [6], where a framework for parsing Indian languages had been presented. After that, Ray et al. [75] Hindi POS disambiguation for local word grouping. Then there was work presented for Tamil language by Arulmozhiet al. [3]. Then a lot of work flourished in other Indian languages as well, including Telugu and Bangla. A lot of these POS taggers have been built using CRFs, such as for Hindi by [82], Bengali by [19], Manipuri by [63], Gujarati by [68] and Kannada by [80].

In Hindi, Shrivastav et al. [82] obtained an average accuracy of 88.95% on a training data of 12,000 tokens which as tagged with 23 tags.

In Bengali, Ekbal et. al [19] obtained an accuracy of 90.3% on training data of 72341 word-forms tagged with IIIT tagset[1] of 26 tags.

In Manipuri, Kishorjit et al. [63] obtained an accuracy of 86.04% on training data of 2400 tokens and test data of 6000 tokens.

In Gujarati, Patel et al. [68] obtained an accuracy of 91.74% on training data of 11185 tokens and test data of 5895 tokens, which was tagged on IIIT Tagset of 26 tags.

In Kannada, Shambhavi et al. [80] obtained an accuracy of 84.58% on training data of 51269 tokens and test data of 2932 tokens, tagged on a tagset of 26 tags.

When it comes to Sindhi language, some work has been done using Perso-Arabic as the preferred script. A rule based POS tagger was developed by Mahar et al. [50]. They developed a lexicon of 26,355 entries and a tagset containing 67 tags. Using both these resources along with about 186 disambiguation rules, their Sindhi POS tagger reported 96.28% accuracy. Later

[1] http://shiva.iiit.ac.in/SPSAL2007/iiit_tagset_guidelines.pdf

on, they improved their tagger by handling ambiguous non-diacritic words using WordNet [51] and increased the accuracy to 97.14%.

On the other hand, we have worked on the Devanagari variant of the script for our work on POS Tagging for Sindhi and a statistical approach for training the tagger. We have described our process in detail in Chapter 4.

Unfortunately, these works could not be replicated or reused in building POS Tagger for Sindhi-Devanagari because these resources are not available publicly. Another reason is that, it has been over 70 years since the partition of India and Pakistan took place and due to influence of Urdu and Punjabi in Pakistan, the Sindhi vocabulary has grown and diverged from the one spoken in India. Sindhi used in India has been influenced by Hindi, Gujarati, Punjabi and other Indian languages. Therefore, these lexicon and resources developed for Sindhi in Pakistan cannot be used directly.

3.2 Transliteration

Transliteration is the process of converting text written in one form of writing (script) into another form of writing. This conversion process is such that the source word and target word have an approximate phonetic equivalence, so that the words are pronounced in roughly the same manner by readers of either scripts.

There are various applications of transliteration models. One of the key applications has been machine translation (MT), where transliteration is primarily used to transliterate the named entities and out-of-vocabulary (OOVs) words of source to target language [33, 18] or build MT systems between closely related languages [17, 62]. Other key application has been cross-lingual information extraction (CLIR) [14, 88, 84, 1], where the goal is to find relevant documents with information in a target script for queries that are expressed in different (source) script. In our case, we are building the transliteration model for levaraging resources in Sindhi Devanagari to Sindhi Perso-Arabic. We shall discuss some common approaches to build transliteration systems in the literature below.

3.2.1 Rule-Based Approach

This approach involves developing a set of hand crafted rules which convert phonemes, characters or sub-strings of an input text in source script to target script. These rules predominantly contain mappings between character sets of both scripts and can also include rules that apply in certain special contexts and cases of failure. This approach is very efficient but faces challenges in cases where one-to-many or many-to-many mappings exist between the characters. This approach has been employed for several languages and orthographies such as Arabic-English [2], Shahmukhi-Gurmukhi (scripts of Punjabi language) [56], English-Korean [66], Hindi-Punjabi [27], Hindi-Urdu [46], Gujarati-Hindi [69], etc. In our work, we have followed this approach to transliterate between Devanagari and Perso-Arabic scripts of Sindhi language.

3.2.2 Machine Learning Approach

Developing a transliteration system by learning a transliteration model is much faster but requires significant amount of training data. In this case, the training data is transliteration pairs. Such pairs are not availble directly and are therefore obtained by extracting from other multilingual resources such as parallel or comparable corpora [83, 81] or from the internet (Wikipedia, web documents) [42, 35, 84, 41].

The transliteration model is then trained using a generative framework [1, 31, 20]. They have been used most commonly for this problem and it involves an algorithm which learns the alignment between each character in both the scripts [65]. Alignment is the most crucial stage which defines the accuracy of the model eventually. This step is equivalent to creating a mapping/rule table between the scripts in a rule-based approach, except that this is an automated process which can also learn various other contextual patterns. Recently, work has been done using discriminative framework by posing transliteration as a classification problem [90, 41]. In this approach, words are paired from comparable corpus and determined whether they its a valid transliteration pair or not.

3.2.3 Transliteration for Indic Languages

Significant research has been done on transliteration for some of the major Indian languages. Since 2009, the Named Entities Workshop (NEWS) has also been organising a shared task on machine transliteration and Indian languages like Hindi, Tamil, Kannada and Bengali have been a part of it. There were many systems that were developed in these shared tasks [48, 49, 91, 92, 4, 16]. The dataset of transliteration pairs had been provided by Microsoft Research India.

Other than English-Hindi transliteration [74, 29], work has also been done on indic-to-indic transliteration. Gupta et al. (2010) [30] worked on WX, a common notation for Hindi, Bengali, Punjabi, Telugu, Malayalam and Kannada. Similarly, Ganapathiraju et al. (2005) [23] had developed Om transliteration scheme, a common script representation for all Indian languages. Chaware et al (2011) [9] worked on rule based Hindi-Marathi transliteration. Malik et al. (2006) [56] worked on transliterating between Gurmukhi and Shahmukhi scripts of Punjabi. Lehal et al. (2010) [46] worked on rule based Hidni-Urdu transliteration while Durrani et al. (2010) [17] employed phrase-based machine transliteration technique to train Hindi-Urdu transliteration and facilitate a translation system between them.

Recently, Kunchukuttan and Pudupally (2015) developed Brahmi-Net [43], which transliterates between languages of the Indian subcontinent. It supports 18 languages and 306 language pairs for statistical transliteration. The supported languages cover 13 Indo-Aryan language (Assamese, Bengali, Gujarati, Hindi, Konkani, Marathi, Nepali, Odia, Punjabi, Sanskrit, Sindhi, Sinhala, Urdu) , 4 Dravidian languages (Kannada, Malayalam, Tamil, Telugu) and English.

When it comes to Sindhi language, there has been some research on transliteration. Leghari and Rahman [45] have presented an approach to transliterate between Perso-Arabic and Devanahari by using an intermediate Roman script. They have not shared any results of their approach but have discussed various issues with these scripts which we had encountered as well. Lehal and Saini [47] have created a hybrid system which combines rules, lexicon, word

and character language models to transliterate the words in Perso-Arabic to Devanagari script for Sindhi, Urdu and Punjabi languages. This system gives an accuracy of 91.68% for Perso-Arabic to Devanagari transliteration. Our work (dicussed in Chapter 6) is completely based on rule-based approach and we focus on Devanagari to Perso-Arabic direction of transliteration.

3.3 Morphological Analysis

Morphology describes the internal structure of words in a language. A morphological analysis of a word involves describing one or more of its properties such as: gender, number, person, case, lexical category, etc. The are two most widely used approaches for developing morphological analysers, which are discussed below.

3.3.1 Rule-Based Approach (Finite State Transducers)

In this approach, the goal of the system is to list all possible analyses of an input word, irrespective of its context in the sentence. This is done by creating finite-state machines using morphological rules and a large lexicon of the language. The morphological rules define the various morphological properties (inflection, derivation, compounding or cliticization) of different word classes (noun, verbs, ajectives, etc.) in the language. The lexicon defines which set of morphological properties each word stem adheres to. These set of rules (also known as paradigms) and lexicon have to be developed manually. If a word that does not exist in the lexicon is provided as input then no analysis could be generated for it. Such words are also called Out-of-Vocabulary (OOV) words.

There are various tools that can be used to develop finite-state transducers such as, OpenFST[2], SFST[3], HFST[4], Xerox's xfst, twolc, and lexc [5] and Apertium's *lttoolbox* [6].

In our work for developing morphological analyser for Sindhi, we have used Apertium's *lttoolbox* [21], which has been used to develop finite state morphological analysers for more than 46 languages.

[2] http://www.openfst.org/twiki/bin/view/FST/WebHome
[3] http://www.cis.uni-muenchen.de/~schmid/tools/SFST/
[4] http://www.ling.helsinki.fi/kieliteknologia/tutkimus/hfst/
[5] https://open.xerox.com/Services/fst-nlp-tools
[6] http://wiki.apertium.org/wiki/Lttoolbox

3.3.2 Machine Learning Approach

In this approach, the linguistics rules describing a language's morphological patterns are mined using supervised or unsupervised machine learning algorithms. Here are some examples.

One of the popular unsupervised approach is that presented by Goldsmith [25], where he used minimum description length (MDL) analysis on European languages to model morphological segmentation. His proposed AutoMorphology tool learns patterns like prefixes and suffixes from a raw corpus. Another popular tool is Morfessor [11], which is a family of probabilistic machine learning methods for finding the morphological segmentation from raw text data.

In case a lexical database with limited information (such as lemma, word form, lexical category tags) is available, supervised learning can used for building morphological analyser. Some examples works using this approach are Bosch [85] and Wicentowski [89].

Machine Learning based approaches can be very helpful in situations where annotated data or linguistic experitse is unavailable for a language. Such a system is good to start with for creating morphological rules but it cannot capture patterns in complex languages (like Arabic). Also, although these appraoches are quick and easy, they are also limited by the requirement of large corpora and are therefore unsuitable for low-resourced languages.

3.3.3 Morphological Analysis for Indic Languages

There has been significant research on morphological analysis for various Indian languages such as Hindi[6, 26, 38, 57], Marathi[5], Tamil [15], Kannada [86], Malayalam[34], Oriya [58], etc. A lot of Indian languages have also been developed using Apertium *lttoolbox* which include Indo-Aryan Hindi, Marathi, Tamil [67], Assamese [71], Oriya [37] and Malayalam [78, 87].

There are only two works available in the literature on morphological analyses for Sindhi. Rahman et al. [73] have worked on capturing morphological construction of Sindhi nouns. Their work investigates Sindhi noun inflection rules and defines equivalent computational rules for creating FSTs (Finite State Transducers).

Jherna Devi [64] has implemented computational resource grammar for Sindhi in Grammatical Framework[7]. Grammatical Framework [76] (abbreviated as, GF) is a functional and natural language processing programming language, which is designed for writing grammars. The Sindhi GF library has around 360 entries in its lexicon. These number of entries belonging to each part-of-speech category is tabulated in Table 5.1. Sindhi grammar library has used different categories and functions to manage the morphology and syntax implementation. The library has 44 categories and 190 functions. Since GF is open-source, we could refer this li-

[7] http://www.grammaticalframework.org/lib/src/sindhi

brary during initial stages of development of our work and we had also found and reported some mistakes in it. For instance, some feminine nouns incorrectly were assigned to a paradigm for masculine nouns.

Both these works were leveraged while laying the foundations of our Morphological Analyzer for Sindhi. The work done by Rahman et al. [73] helped in understand Sindhi noun morphology and developing paradigms for it. On the other hand, the work done by Jherna Devi [64] in GF was available open-source and helped us in building the initial lexicon for various lexical categories and also getting an idea for how we could construct paradigms for lexical categories other than nouns.

3.4 Other works in NLP for Sindhi

Apart from morphology and POS Tagging, researchers have also contributed towards other aspects of natural language processing for Sindhi. Mahar et al. have worked on various tasks such as text segmentation [54], language modeling for word prediction [53], lexicon based diacritic restoration [52], text-to-speech synthesis [55], etc.

Pinkey Nainwani [59] [60] has done extensive research to develop Machine Translation (MT) system for English - Sindhi language pair. The parallel corpus between English and Sindhi-Devanagari developed during her PhD work [61] was leveraged by us in developing the Sindhi POS tagger.

Chapter *4*

Part of Speech Tagging for Sindhi

Languages around the world have a common feature of syntactic ambiguity. An example from English language, as explained in [22] is the word *left*. We can observe that in different situations it functions as an adjective (as in the left mouse button), an adverb (turn left), a noun (I was sitting to his left) or a verb, either a past tense form (he left his wife) or a past participle (she was left wondering). Part-of-Speech (POS) tagging is the process of resolving such ambiguity by annotation of lexical categories. POS tagging assigns an appropriate part of speech tag for each word in a sentence of a natural language. An automatic Part-of-Speech tagger can be developed if a large annotated data corpus or a comprehensive set of linguistically motivated rules is present.

The importance of POS tagged data is enormous in language processing, language technology and development. The main goal of a POS tagging is to disambiguate a word and assign it to appropriate part-of-speech category. POS tagged data serves as key input in various NLP applications such as machine translation, information retrieval, natural language parsing, etc. Although we know the importance of this resource, not much effort has been put into developing it for Sindhi.

In this chapter, we present our work on collection of raw corpus for Sindhi-Devanagari, creating a POS annotated dataset from it and then developing a CRF based POS tagger. We have discussed the various features used incrementally for developing the tagger.

4.1 Corpus Creation

A sufficiently large annotated corpus is required to build a statistical POS Tagger, which had not been previously developed as the amount of raw text available on the web for Sindhi-Devanagari was very less. The problem was further compounded as many publishers on the

web have not yet moved to Unicode standards. We contacted various publishers and news agencies to source raw data which could be annotated with POS tags. Some blogs available online had to be discarded as they often contained ungrammatical structures or un-naturally long sentences. Eventually, most of the data collected was manually typed with Unicode encoding for Devanagari and annotated with POS tags. This manual process allowed us to remove words in different scripts and grammatically incorrect sentences. We picked up stories, articles, news, general conversation from the web to create our corpus. We now have a raw corpus of 326813 words, with average sentence length of 9.3 and a vocabulary (unique words) of 22300 words.

4.1.1 The Annotation

To start with this task, we did not have an annotation scheme and tagset defined for Sindhi. So, we adapted an existing BIS (Bureau of Indian Standards) tagset.

The BIS Tagset[1] has been designed under the banner of Bureau of Indian Standards. This POS schema is based on W3C XML Internationalisation best practices. The BIS Tagset contains the features of a hierarchical tagset. However, it has tags for only first two tiers of linguistic information (POS and their subtypes) and excludes information from tier three onwards as these can be provided by morph analyzers and parsers. The BIS Tagset is comprehensive and is designed to be extensible to any Indian Language[2].

The original tagset has 38 tags. Besides adjective, adverb and postposition all other categories have some further distributions. There are certain tags which were not included for Sindhi, such as Quotative Conjunction (CC_CCS_UT) , Verbal Nouns (N_NNV) and some SubPOS level-2 forms of Verbs, that is Finite, Non-Finite, Infinitive and Gerund (V_VM_VF, V_VM_VNF, V_VM_VINF & V_VM_VNG). The final adapted tagset has 32 labels (see Appendix Table A.1).

We started annotating the sentences in this corpus using POS tags from the adapted BIS Tagset, with the help of two annotators. Currently, we have tagged 44692 words. We used them to build an automatic Part-of-Speech Tagger, which we will discuss in Section 4.3.

4.1.2 Inter-Annotator Agreement

The meaning of a sentence and its words can be interpreted in different ways by different readers. This subjectivity can also reflect in annotation of sentences of a language despite the

[1]The tagset does not have a standard published reference yet. It is under the process of being accepted as a standar. We refer to the document circulated in the consortia meetings : "Unified Parts of Speech (POS) Standard in Indian Languages"

[2]http://www.tdil-dc.in/tdildcMain/articles/134692Draft%20POS%20Tag%20standard.pdf

annotation guidelines being well defined. Therefore, inter-annotator agreement is calculated to give a measure of how well the annotators can make the same annotation decision for a certain category. Through this measurement we are able to conclude the following things:

1. How well defined is the annotation scheme? It will make it easier to classify a word into its category and annotators will agree on that classification.

2. How ambiguous are words in the language ? Annotators will find difficulty disambiguation of a word into a category and it will lead to more disagreement.

3. What is the quality of corpus? If sentences in the corpus are simplistic then agreement would be more as annotation is easier, despite quality of annotation scheme or characteristics of the language.

4. How well versed are the annotators themselves. Despite the adversities in annotation scheme and/or the data of language, if the annotators are well versed with both of them, the disagreement would be less likely.

Cohen's Kappa [10] is one of the most common metrics for measuring inter-annotator agreement. The range of Kappa is from -1 to +1 , where 0 represents agreement due to random chance, 1 implies perfect agreement between raters.

$$\kappa = \frac{p_o - p_e}{1 - p_e} = 1 - \frac{1 - p_o}{1 - p_e}$$

where p_o is the proportion of observed agreement and p_e is the proportion in agreement due to chance.

We took 793 words which were annotated by both the annotators. We calculated Cohen's Kappa on it and obtained a score of 0.93 . The pairs with most disagreement were (NNP, NN), (NN, JJ) and (DMI, DMQ).

4.2 Conditional Random Fields

Conditional Random Fields (CRFs) [44] is a form of undirected graphical model that defines a single log-linear distribution over label sequences given a particular observation sequence. CRFs calculate conditional probabilities of values on the output nodes, given values of input node in an undirected graph. So, the conditional probability of a state sequence $S =< s_1, s_2, \ldots, s_T >$ given an observation sequence $O =< o_1, o_2, \ldots, o_T >$ is calculated as

$$P_\Lambda(s|o) = \frac{1}{Z_o} exp(\sum_{t=1}^{T} \sum_{k} \lambda_k f_k(s_{t-1}, s_t, o, t))$$

where $f_k(s_{t-1}, s_t, o, t)$ is a transition feature function of the entire observation sequence, whose weight λ_k is to be learned via training. The values of the feature functions may range between $-\infty \cdots + \infty$, but typically they are binary.

To make all conditional probabilities sum up to 1, we must calculate a normalization factor

$$Z_o = \sum_{s} exp(\sum_{t=1}^{T} \sum_{k} \lambda_k f_k(s_{t-1}, s_t, o, t))$$

which, as in HMMs, can be obtained efficiently by dynamic programming.

The objective function to be maximized for training CRF is the penalized log-likelihood of state sequences given observation sequences :

$$L_\Lambda = \sum_{i=1}^{N} log(P_\Lambda(s^{(i)}|o^{(i)})) - \sum_{k} \frac{\lambda_k^2}{2\sigma^2}$$

Where, $\{ < o^{(i)}, s^{(i)} > \}$ is the labeled training data. The parameters λ here are set to maximize the penalized log-likelihood using Limited-memory BFGS [79].

The primary advantage of CRF over HMMs is the conditional nature, resulting in relaxation of independence assumption required by HMMs (Hidden Markov Models). CRF also avoid the label bias problem of Maximum Entropy models and on other directed graphical models. Thus, CRFs outperform HMM and ME models on a number of sequence labeling tasks. [44, 79, 70]

For the case of POS tagging, an observation sequence is tokens of a sentence and the state sequence is its corresponding sequence of labels or POS tags. CRFs can be applied to a variety of NLP tasks, such as Named Entity Recognition, Information Extraction, Text Chunking, etc [44]. We have used CRF++[3] for training and testing our tagger.

[3] http://taku910.github.io/crfpp/

4.3 Experiments

In this section, we have described the various experiments that we performed for training a good model. We carried out several experiments with different set of features, analysed our results and then introduced new features based on the error analysis. We have also described the features in order with the corresponding experiment they were introduced in. The trained models were evaluated in the following manner. We included the complete tagged corpus and calculated 10-fold cross-validation accuracy and best accuracy.

$$\text{Accuracy} = \frac{\text{Total no. of correctly tagged tokens}}{\text{Total no. of tokens}}$$

$$\text{Average Accuracy} = \frac{\text{Accuracy of all folds}}{\text{Total no. of folds}}$$

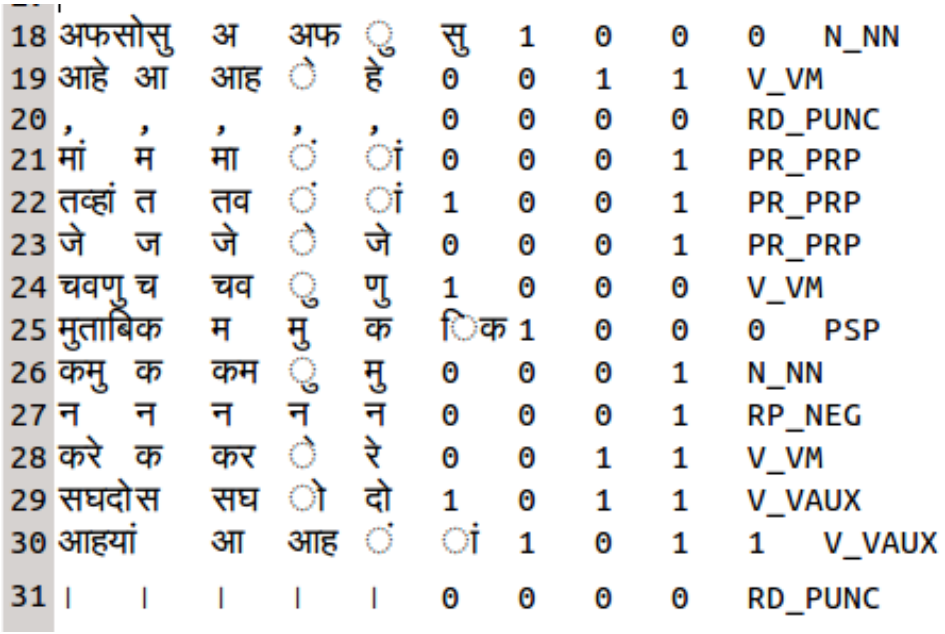
4.3.1 CRF-0, The Simple Baseline

We first created a simple model without using any features. In this model, we would predict the best tag for a given input word based only on the frequency of their (word and tag) occurrence together in the training data. This model helped us in understanding what one could achieve with this amount of annotated data and without any linguistic knowledge to incorporate as features in training a POS tagger. Since there is no actual model training and feature learning involved, we refer to this model as the simple baseline. This experiment resulted in a model with best case (out of 10-folds) accuracy of 82.35% and an average accuracy of 80%. Thus, features are important and we need them to build a better model with reasonable accuracy.

4.3.2 CRF-1, Incorporating Contextual Features

A renowned linguist, Professor J. R. Firth, has said *You shall know a word by the company it keeps*. Context (preceding and following tokens with respect to a token) is one of the most important and fundamental property that helps in resolving the syntactic ambiguity of a word in a phrase or sentence. In this experiment, we used context and combination as our only features to train this model.

A context window of 5 is represented as "[-2,2]" and it implies 5 uni-grams : tokens whose position is in the range of of -2 to 2 relative to the current word and the current word itself. Combination on the other hand combines the next and the current token into one token (represented as : 0/1). The bi-gram feature (represented as : B) is also a combination feature, it creates a set of unique features from the all the features of the current and the previous tokens.



18	अफसोसु	अ	अफ	◌ु	सु	1	0	0	0	N_NN
19	आहे	आ	आह	◌े	हे	0	0	1	1	V_VM
20	,	,	,	,	,	0	0	0	0	RD_PUNC
21	मां	म	मा	◌ं	◌ां	0	0	0	1	PR_PRP
22	तव्हां	त	तव	◌ं	◌ां	1	0	0	1	PR_PRP
23	जे	ज	जे	◌े	जे	0	0	0	1	PR_PRP
24	चवणु	च	चव	◌ु	णु	1	0	0	0	V_VM
25	मुताबिक	म	मु	क	◌िक	1	0	0	0	PSP
26	कमु	क	कम	◌ु	मु	0	0	0	1	N_NN
27	न	न	न	न	न	0	0	0	1	RP_NEG
28	करे	क	कर	◌े	रे	0	0	1	1	V_VM
29	सघदोस	सघ	◌ो	दो	1	0	1	1	V_VAUX
30	आहयां	आ	आह	◌ं	◌ां	1	0	1	1	V_VAUX
31	।	।	।	।	।	0	0	0	0	RD_PUNC

Figure 4.1: A sample sentence with features (2 suffixes, 2 prefixes, few binary features) and POS Tag for every word.

This model gave best case (out of 10-folds) accuracy of 90.12%. We have seen that context has been used for POS tagging since the beginning. The context can only get us so far but the specialty of Indo-Aryan languages is their morphological richness, which we shall consider in the next experiment.

4.3.3 CRF-2, Incorporating Affix Features

Morphology is the study of the internal structure of words. However in the absence of a morphological analyser for Sindhi Devanahari, we decided to consider them heuristically by using affix features. Affixation is a process which defines morphology of a word by attaching an affix to its root form. Prefixes and Suffixes are two kinds of affixes which we made use of to capture the morphology of the Sindhi words. We included first and last few characters of a word as its features in the data. An example is given in Figure 4.1, where the first two columns represent prefixes upto length 2 and next two columns represent suffixes upto length 2. We experimented with various combination of number of prefixes and suffixes and eventually got the best case accuracy of 93.73% by using 3 prefixes and 6 suffixes.

4.3.4 CRF-3 and CRF-4,Incorporating Lexical Features

Words in a language are broadly classified into open class and closed class. Open class (nouns, verbs, etc.) use inflection quite extensively which makes them lengthy. We thought including word length as a feature might help in discriminating between open and closed classes.

A binary feature for word length was added to the training and testing data. Its feature value was set to 1 if the length of a word exceeded 3 characters, else 0.

We noticed that 8.5% of Numerals (tagged as QT_QTC) were being miss-classified as Nouns. In our data we have digits in either of these closed sets : Roman [0-9] or Devanagari ०-९ script. We can use this information to help classify the numerals better. So, we introduced binary feature for indicating whether the token is numeral (in CRF-4).

4.3.5 CRF-5 and CRF-6, Incorporating Categorical Features

We had catered to open class words earlier by incorporating their inflectional property. Similarly, we could also cater to closed class words (or function words). Function words occur with open class words in their context. There are also instances where a similar context may not necessarily mean the presence of a closed class word. This can be a source of ambiguity. Another property of function words is that they have a very high frequency in the corpus. These properties could be used to make classification of function words better.

We created a list of top 150 high frequency words from the corpus. We used this list to include another binary feature, indicating if the word is a possible function word (based on this frequency list). This configuration of *CRF-6* produced the best model so far for POS tagging Sindhi-Devanagari text, with average accuracy of 91.78%.

The results of all the experiments are consolidated and reported in Table 4.1. We also calculated the accuracies of unknown words (in the testing data) with every experiment. Figure 4.2 reports the comparison between accuracy numbers of unknown words and test data.

4.4 Error Analysis and Observations

We observed through our experiments that incorporating contextual and morphological features gave us maximum gains, 7.73% and 3.88% in average accuracy respectively. Although including lexical and categorical features did not change the average accuracy by large amount but caused significant impact to specific categories like QT_QTC (quantifier) and V_VAUX (auxiliary verbs).

Model	Context	Affixes	Comb.	WL	NUM	AUX	FW	Accuracy	Avg. Accuracy
CRF-0	-	-	-	-	-	-	-	82.35	80.00
CRF-1	[-1,1]	-	0/1	-	-	-	-	90.12	87.73
CRF-2	[-1,1]	3,6	0/1 ; B	-	-	-	-	93.73	91.61
CRF-3	[-1,1]	3,6	0/1 ; B	yes	-	-	-	93.80	91.7
CRF-4	[-1,1]	3,6	0/1 ; B	yes	yes	-	-	93.85	91.73
CRF-5	[-1,1]	3,6	0/1 ; B	yes	yes	yes	-	93.95	91.75
CRF-6	[-1,1]	3,6	0/1 ; B	yes	yes	yes	yes	94.01	91.78

Table 4.1: Accuracy of each model and the features it was trained on. Context = range of adjacent tokens considered. Affixes = (prefixes, suffixes). Comb. = Combination features. WL = Word length. AUX = Auxiliary verbs. FW = Function words.

Actual	Predicted	Count
JJ	N_NN	38
V_VM	V_VAUX	20
N_NNP	N_NN	17
N_NN	JJ	17
V_VAUX	V_VM	15

Table 4.2: Top 5 hard-to-disambiguate pairs

In Table 4.2 we have the most common ambiguities, that our best model could not resolve. Similar ambiguities have been reported for the same task in other Indian languages as well. We can resolve this further by decreasing granularity of the tagset or by using using resources such as NER and Lexicons. We currently don't have any such resources.

The experiments above have shown the effect of various features on the model accuracy. An interesting fact that we noted was the effect of training data size on model accuracy. So, we compared our baseline and best model on the basis of size of training data and also observed the curve of accuracy versus data size (see Figure 4.3). We observed that the curve has not reached a plateau, eventually. This suggests that there is scope of further improvement in accuracy by using more training data.

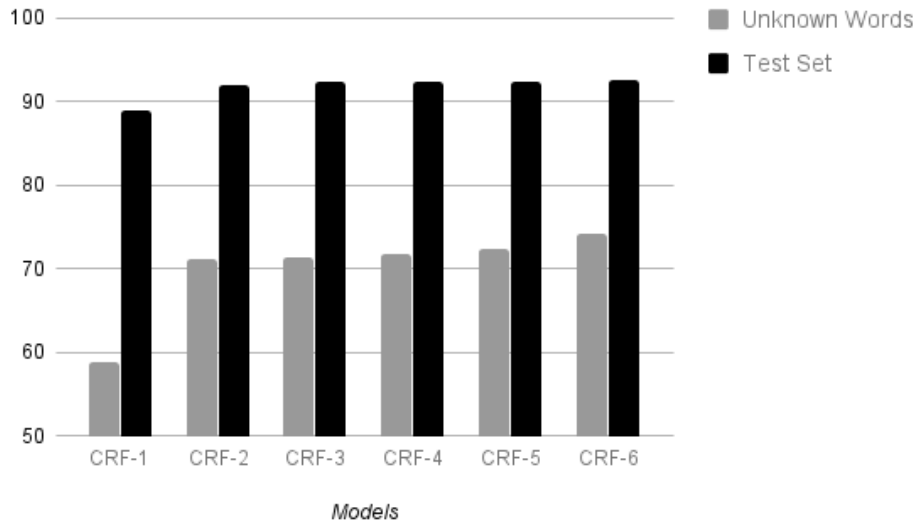


Figure 4.2: Accuracy on Test Data and Unknown words.

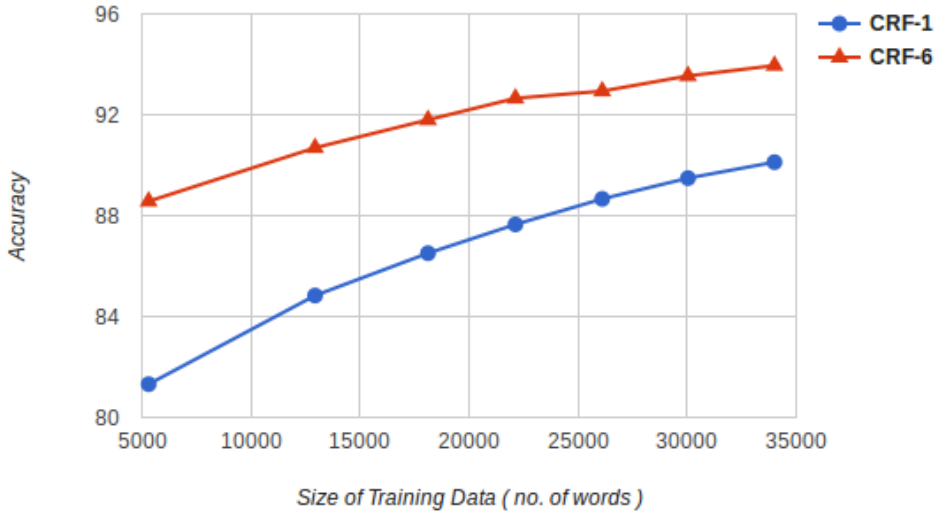


Figure 4.3: Plot of Accuracy v/s Size of Training Data, comparing the baseline and best model.

4.5 Conclusion

A POS annotated corpus for a language is a very useful resource for a language in the field of language technology and processing. We have built some basic resources for computational linguistics in Sindhi. This includes a manually created dataset, a POS annotated corpora and automatic POS Tagger.

We have reported the accuracy obtained by training a CRF based model on POS annotated data of a less resourced language Sindhi. We found that using linguistically oriented features (affixes, word length, stop words, auxiliary verbs) makes a significant impact. These features coupled with capturing context help in developing a good POS-Tagger. Then, the size of training data is the next most important factor in further enhancing the tagger. Currently, the size data (labelled as well as uinlabelled) in Sindhi Devanagari is very small. Manual creation of raw data is an expensive task and cannot be scaled easily to create large datasets and conduct various experiments.

Chapter *5*

Morphological Analyser for Sindhi

The study of words and how they are formed in a language is called Morphology in linguistics. Morphology describes the internal structure of words in a language. The vocabulary of a language consists of a lot of words which are formed by combination of a limited set of morphemes. These morphemes are individually meaningful units as they indicate various properties of a word.

Morphological analysis involves breaking down a word into morphemes and describing one or more of its linguistic properties such as: gender, number, person, case, lexical category, etc. Morphological analysis of a word thus becomes one of the fundamental tasks in natural-language processing for a language. It is a vital resource in building NLP tools such as spell-checkers, parsers and are essential linguistic features in applications like information retrieval, machine translation, etc.

In this chapter, we present our work on developing a morphological analyser for Sindhi Perso-Arabic. We decided to work on Perso-Arabic for various reasons but the most predominant reason was availablity of large corpus in Sindhi Perso-Arabic. A large corpus covers various words and word forms that are present in the language which are important for better and reliable language understanding. Such kind of coverage was not available in the small datasets created for Sindhi Devanagari.

5.1 Sindhi Morphology

Sindhi, like many Indo-Aryan languages, is a morphologically rich language. It uses suffixes for constructing derivational and inflectional morphemes. The words in language can be classified into primary and secondary words. The primary words are simple, indivisible, free morphemes. Whereas, secondary words are divisible into compound (combination of two or

more primary words) and complex words (addition of prefixes or suffixes). We have described certain aspects of Sindhi morphology in various lexical categories in the following subsections.

5.1.1 Nouns

Sindhi nouns are marked by a gender. The words in Sindhi language generally end in a vowel, which (in case of nouns) also help them classify into their appropriate gender. There are two genders in Sindhi : masculine and feminine. Feminine nouns generally end with the following vowels: اَ [ə], ا [a] and اِ [i] and the masculine nouns usually end with و [o] or اُ [u]. Nouns also inflect according to number (singular and plural) and case (nominative and oblique). Nominative is the default case without any inflection and nouns in oblique case are generally followed by a postposition.

Pronouns, like nouns, also inflect with gender and number. Pronouns are a closed category but may be categorized into several subcategories: personal, demonstrative, indefinite, interrogative, reflexive, relative and co-relative.[1]

5.1.2 Adjectives

Adjectives can be classified into two main classes, declinable (سنو *suṭho* 'good') and indeclinable (آسان *āsan* 'easy'). All adjectives ending in و [o] are declinable and show agreement in gender, number and case with the following noun.

Adjectives have three degrees for comparison: analytical positive, comparative (هُن کان وڌو *hun khā vado* 'older than him') and superlative (سڀ کان وڌو *sabh khā vado* 'the oldest').

5.1.3 Verbs

Verbs are morphologically the richest, most complex and largest category of all. They can be marked by number, gender, person case, tense, aspect and mood. Here are some properties and kinds of the Sindhi verbs.

The auxiliary verbs modify the action expressed by the main verb. They constitute a small class of words. An example of auxillary is سگ *səgh-* 'be able' and copula is هو *ho-* 'be'.

Sindhi main verbs also exhibit transitivity. An example with a transitive verb would be : مان کت لکان تو *mã khat lIkhã tho* 'I write a letter'. Similarly, an example with an intransitive verb : مان سمهان تو *mã sumhã tho* 'I am sleeping'. The distinction between transitive and

[1] A co-relative pronoun is a feature of some Indo-Aryan languages where a relative pronoun in the relative clause has a counterpart (the co-relative) in the main clause.

intransitive verbs is important for morphological disambiguation and parsing as the transitivity of the verb determines the subject case in certain tenses.

5.1.4 Adverbs

Adverbs in Sindhi are indeclinable (uninflected). They decline only when other lexical categories (nouns, adjectives) are used as adverbs and show the same inflectional properties of the original category.

5.1.5 Postposition

Postpositions are functional words which are used to show grammatical relations. They are mostly indeclinable (uninflected) with some exceptions, such as جو *jo* 'of', which is a possessive marker and inflects to agree in gender and number with possessed noun.

5.1.6 Conjunctions

Conjunctions are indeclinable (uninflected) in Sindhi. These are further classified into two categories: coordinating conjunctions and subordinating conjunctions. Coordinate conjunctions are either *cumulative*, which add one statement to another (ءِ *ain* 'and'), or *alternative*, which express a choice between two alternatives (يا *ya* 'or'). Subordinate conjunctions join subordinate clauses to construct a complex sentence. They may also at times express time, location, direction, manner, reason, condition, result, concession etc. (جڏهين - تڏهن *jadhein - tadhein* 'when - then').

5.1.7 Interjections

Interjections are words or phrase which expresses some sudden feeling or emotion. Some examples are: واقعي *wakaI* 'Really!' , واهه *wah* 'Wow!'.

5.2 Developing The Morphological Analyser

We initiated our work on developing Sindhi morphological analyser with the help of three resources. The first one was an article by [73], which described how Sindhi nouns inflect. This aided us in creating our first few paradigms for nouns.

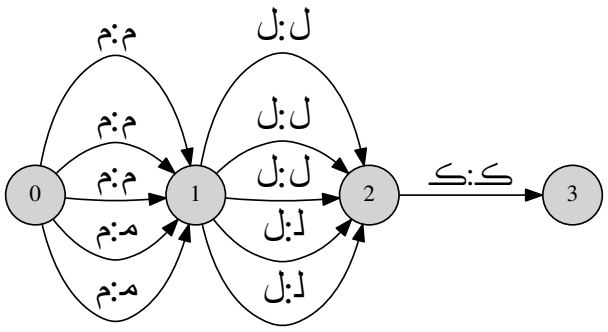


Figure 5.1: Example of a fragment of a transducer for ملک *mulk* 'country' demonstrating how badly encoded text is dealt with. The left side is the output and the right side is the input. Note that the ک [k] character also has initial, medial and final forms, but these are produced with a separate code point, ـ (U+0640).

The second resource was Sindhi GF library. It helped us in verifying some of the paradigms that we had already defined. It was also helpful in adding verb and pronoun paradigms and improve the paradigms for nouns. The third resource was a corpus, a collection of articles from Sindhi Wikipedia. Then, we used our knowledge of Sindhi, the Wikipedia corpus and lexicon from GF to develop a lexicon.

The process of adding words to lexicon was completely manual. We had parsed the corpus to create a list of words with their frequency, sorted in descending order. We went through this list word by word and added each word we knew to the lexicon along with the corresponding paradigm that it belonged to. An example paradigm and entry can be found in Figure 5.2. Also, an example sentence and its morphological analysis produced by Apertium is shown in Figure 5.3.

We referred to dictionaries[2] and grammar books[3] as well, that were available online or in printed hard copy form, to add more words and paradigms. We also tried crowd-sourcing for understanding words that we could not find anywhere, by asking learned people through social media. The current statistics of words in lexicon are tabulated in Table 5.1, also drawing a comparison with GF lexicon. There are total 72 paradigms in our analyser right now.

[2] http://www.sindhila.org/dics.php?dic=sindhidevnagarienglishdic

[3] http://www.ciil-lisindia.net/Sindhi/sindhi_struct.html and http://www.sindhila.org/sindhilearning/

```
Paradigm:
<pardef n="و/چوکر__n_m" c="I">
 <e><p>
  <l>و</l> <r>و<s n="n"/><s n="m"/><s n="sg"/><s n="nom"/></r>
 </p></e>
 <e><p>
  <l>ي</l> <r>و<s n="n"/><s n="m"/><s n="sg"/><s n="obl"/></r>
 </p></e>
 <e><p>
  <l>ا</l> <r>و<s n="n"/><s n="m"/><s n="pl"/><s n="nom"/></r>
 </p></e>
 <e><p>
  <l>ن</l> <r>و<s n="n"/><s n="m"/><s n="pl"/><s n="obl"/></r>
 </p></e>
</pardef>
Entry:
 <e lm="چوکرو"><i>چوکر</i><par n="و/چوکر__n_m"/></e>
```

Figure 5.2: An example paradigm `pardef` and entry `e` for the noun چوکرو *chhokro* 'boy' in XML format.

Part of speech	**Number of stems**	
	Apertium	*GF*
Noun	1191	179
Verb	88	54
Adjective	766	49
Proper noun	958	1
Adverb	267	21
Numeral	52	4
Conjunction	17	7
Interjection	7	1
Abbreviation	6	0
Postposition	66	21
Pronoun	23	14
Determiner	13	10
Total:	3454	361

Table 5.1: Number of stems in each of the categories in the Apertium and Grammatical Framework lexica.

5.2.1 Orthographic Challenges

Many computational approaches to Indian languages use a transliterated representation of the language for the lexicon and morphology, for example the popular WX notation [6]. This has the disadvantage that in order to use the lexicon on real-world text it is necessary to convert to/from the transliterated representation.

We use Unicode as a character set for our lexicon as this is a global standard. However, when working with Sindhi and other similar writing systems it presents a number of issues:

1. One letter may have many forms, all of which have separate Unicode code points: isolate, initial, medial, final. For example, the letter مر [m] (U+0645 in its canonical form) may appear as مر (U+FEE3, initial), ـمـ (U+FEE4, medial), ـم (U+FEE2, final) or م (U+FEE1, isolate). In most text these specific presentation forms do not appear, as the choice between them is determined by the layout software.

2. Since Sindhi shares its script with Persian, Arabic and Urdu , there are a lot of character homophones that get introduced into Sindhi. One example is the letter 'h' : ه U+0647 (in Sindhi) and ھ U+06BE (in Urdu). Both these letters are used interchangeably in the text.

3. A lot of Perso-Arabic script based languages do not use diacritics marks in their texts. This creates several issues:

   (a) Semantic ambiguities: Words may have multiple interpretations when used without diacritics. For example: ملڪ *mlk* can be either *mulk* 'country' or *milk* 'milk'.

   (b) Syntactic ambiguities: Sometimes presence of diacritics changes not only the meaning but lexical category as well. For example: هو *ho* can be verb 'was' or pronoun 'that'.

   The practice of using or not using diacritics in text is not standard among writers. A lot of texts contain both kinds of words. This created problem for us as many times our analyser could not analyse a word (with diacritics) despite the correct analysis of that word (without diacritics) was present in the lexicon.

In order to get around the problem of analysing badly or incorrectly encoded Unicode text, for each canonical letter, we allow the other code points as variants on the input side of the transducer (shown in Figure 5.1). We also do the same for character homophones.

^ڈیاري/ڈیاري<n><f><sg><obl>$

^جي/جي<post><m><sg><obl>$

^موقع/موقعي<n><m><sg><obl>$

^تي/تي<post>$

^ماٹھو/ماٹھو<n><m><pl><nom>$

^دکانن/دکان<n><m><pl><obl>$

^۽/۽<cnjcoo>$

^گھر/گھرن<n><m><pl><obl>$

^جي/جي<post><f><sg><obl>$

^صفائي/صفائي<adj><f><sg><obl>$

^کرڻ/کندا<vblex><tv><pres><hab><p3><m><pl>$

^هو/آهن<vbser><pres><p3><pl>$

Figure 5.3: Example output for the sentence :

ڈیاري جي موقعي تي ماٹھو دکانن ۽ گھرن جي صفائي کندا آھن

dyarī jey maukey tey manhoo dukānan ain gharan jī safaI kandā āhin

"On the occasion of Diwali, people clean shops and houses."

5.3 Evaluation

We have evaluated the morphological analyser in two ways. The first was by calculating the naïve coverage and mean ambiguity on freely available corpora (shown in Table 5.3) and the second was by calculating precision and recall. Naïve coverage refers to the percentage of surface forms in a given corpora that receive at least one morphological analysis. Although, forms counted by this measure may have other analyses which are not delivered by the transducer. The mean ambiguity measure was calculated as the average number of analyses returned per token in the corpus.

5.3.1 Corpus

There are a lot of websites, books, blogs, etc. on the internet which can serve as sources of raw text in Sindhi. Our primary source of text data was Wikipedia, which fortunately exists for Sindhi language too. We will describe how we sourced the data below.

The collection of articles published on Wikipedia for all the languages available in the forms of compressed dumps on a website[4]. These dumps are updated on a regular basis. We downloaded a Sindhi Wikipedia dump[5] and created our corpus in the following manner.

1. We decompressed the `sdwiki-20150826-pages-articles.xml.bz2` file, which gives us all the articles in XML format.

2. We extracted the raw text from compressed file itself by using a script.[6] The extracted raw text was of size 2.6MB.

3. We noticed there were still a lot of problems in the data. For instance, there was XML metadata and portions of non-Sindhi (English, Urdu, Arabic, Persian) texts. So, we cleaned it manually and eventually got 2.5MB of Sindhi text.

4. The final text has 303,401 words while the filtered out non-Sindhi text had 7,570 words.

We also gathered more textual data by scraping from various domains[7] on the web, as done in [72]. These include texts from news articles (politics, current affairs, sports, editorials), blog posts, forum posts, etc. The size of this collection is about 6.4 MB, with about 805,000 words.

[4] `https://dumps.wikimedia.org/`

[5] `https://dumps.wikimedia.org/sdwiki/sdwiki-20150826-pages-articles.xml.bz2`

[6] `http://wiki.apertium.org/wiki/Wikipedia_Extractor`

[7] `http://svn.code.sf.net/p/apertium/svn/incubator/apertium-snd/URLs.txt`

	Precision (%)	Recall (%)
Known tokens	97.68	97.52
All tokens	97.68	72.61

Table 5.2: Precision and recall presented as percentages

5.3.2 Precision and Recall

Precision and recall are measures of the average accuracy of analyses provided by a morphological transducer. Precision represents the number of the analyses given for a form that are correct. Recall is the percentage of analyses that are deemed correct for a form (by comparing against a gold standard) that are provided by the transducer.

To calculate precision and recall, it was necessary to create a hand-verified list of surface forms and their analyses. We extracted 1000 unique surface forms at random from the Wikipedia corpus, and checked that they were valid words in the languages and correctly spelled. Where a word was incorrectly spelled or deemed not to be a form used in the language, it was discarded.

This list of surface forms was then analysed with the analyser, and each analysis was checked. Where an analysis was erroneous, it was removed; where an analysis was missing, it was added. This process gave us a 'gold standard' morphologically analysed word list of 384 forms. The list is publicly available for each language in Apertium's SVN repository.

We then took the same list of surface forms and ran them through the morphological analyser once more. Precision was calculated as the number of analyses which were found in both the output from the morphological analyser and the gold standard, divided by the total number of analyses output by the morphological analyser.

Recall was calculated as the total number of analyses found in both the output from the morphological analyser and the gold standard, divided by the number of analyses found in the morphological analyser plus the number of analyses found in the gold standard but not in the morphological analyser. The results for precision and recall are presented in Table 5.2.

5.3.3 Qualitative evaluation

We manually analysed the output for error analysis and found the following problems :

1. **Diacritics**: When the diacritised input is given, it is difficult to lookup in the lexicon and disambiguate.

Corpus	Tokens	Coverage (%)	Mean ambig.
Wiki.	341.5k	81.12	3.2
Blogs	805k	76.68	3.4
Average	–	78.90	3.3

Table 5.3: Results of naïve coverage tests.

2. **Miscategorization**: Some stems were assigned to wrong categories or paradigms. For example, لکڻ *likhan* 'to write' was initially marked as an intransitive verb and later corrected to transitive. Such mistakes also existed in GF lexicon.

3. **Incomplete Paradigms**: Some paradigms were insufficient or incorrect. For example, the suffix ءَ [ə] that is attached to some nouns and proper nouns in oblique cases was missing in the paradigms earlier.

4. **Size of Lexicon**: Although the coverage is greater than 80%, the lexicon is still small. Therefore, some of the random surface forms selected for evaluation had unanalysed output.

5.4 Conclusion

We have presented the first freely available morphological analyser for Sindhi. The analyser is based on a word-and-paradigm model of Sindhi morphology and is implemented as a finite-state transducer which can also be used for generation. The lexicon is entirely encoded in Unicode and has reasonable coverage for few lexical entries. The precision of the paradigms is good, as would be expected from a manually constructed resource, but the recall is limited by the small size of the lexicon.

Chapter *6*

Transliteration

Devanagari and Perso-Arabic are the two of the most frequently used scripts to write in Sindhi language. These writing systems are significantly different from each other and are mutually incomprehensible. We had discussed earlier in Section 1.1 that the amount of digital content available on the internet for Sindhi in Perso-Arabic is a lot more than that in Devanagari. Thus, there is a dire need for a system to unite the diversity in scripts of the same language.

The process of converting text from one writing system (script) to another, based on phonetic equivalence is called transliteration. Given the situation with scripts of Sindhi, a transliteration system is a good solution and a much needed tool to bridge the gap between Perso-Arabic and Devanagari scripts. Moreover, such a system would facilitate sharing of resources developed in either of the scripts.

In this chapter, we shall describe the challenges faced, our approach , experiments and results in transliterating from Sindhi in Devanagari to Perso-Arabic script. We shall also describe our experiments in leveraging either scripts. One such experiment was using our previously developed Part-of-Speech tagger in Devanagari script and build a tagger for Sindhi-Perso Arabic by transliterating the tagged corpus. Another experiment was leveraging the Sindhi Devanagari POS tagger by providing more unlabeled data and bootstrapping the tagger.

6.1 An Overview of Sindhi Scripts

6.1.1 The Perso-Arabic Script

The Perso-Arabic script is composed of 52 letters, including Persian letters, digraphs and eighteen other letters (illustrated in Table 6.1) to capture the sounds particular to Sindhi and other Indo-Aryan languages. This script is an abjad script and is written from right to left.

ڳ	[ŋ]	ڃ	[ɲ]	ٻ	[ɓ]	ڱ	[ɠ]	ڄ	[ʃ]	ڀ	[bʱ]
ڪ	[k]	ڇ	[cʰ]	ڈ	[dʱ]	ڻ	[ɳ]	ٽ	[ʈʰ]	ڈ	[ɗ]
ق	[pʰ]	ت	[ʈ]	ڊ	[ɖ]	ڑ	[ɽ]	ٹ	[tʰ]	ڈ	[ɖʱ]

Table 6.1: The characters in the Sindhi alphabet which are not found in the Persian alphabet and their phonetic value.

Each letter has four forms based on its position (beginning, middle, end and standalone) in the word. There are 3 main vowels ا, و and ي which are also treated as consonants when they occur at the beginning of the word. In the middle position they maybe consonant or vowel, depending upon the context. The script also has diacritical marks, which are similar to dependant vowels of Devanagari. Since its an abjab script, the usage of diacritics is optional.

6.1.2 The Devanagari Script

The Devanagari script is an abugida where each character is represents syllable. This is written from left to right. Sindhi Devanagari comprises of 65 leter, which includes vowels, consonants and 4 special Sindhi implosives (ग़, ज़, ड़ and ब़). The vowels are of two types, dependant and independant. The dependant forms are used in the beginning of the word, while the independant vowels are used in the middle and end. Some exceptions have dependant vowels in the middle. Unlike Perso-Arabic, usage of dependant vowels is not optional in Devanagari.

6.2 Challenges in Sindhi

There are many challenges that we faced for this task. We shall described them below.

1. **Unavailability of Transliteration Pairs** : Transliteration pairs is a key resource for learning a transliteration model. In cases where a seed set is not available, transliteration pairs can be easily mined from a parallel data between the source and target language pair. We do not have large parallel texts between Sindhi (Perso-Arabic) and Sindhi (Devanagari). Unfortunately, this rules out the possibility of using machine learning based approaches for Sindhi transliteration.

2. **Missing Diacritics** : Perso-Arabic script based languages do not use diacritics in their texts, which are used to represent short vowels. The readers of this script are able to

decode the word in context and understand its correct meaning. This is relatively easy for human readers than a machine as the absence of short vowels would create semantic and syntactic ambiguities due to possibilities of multiple interpretations of the same word. An example: '**چپ**' *cp* can be either *capa* 'lips' or *cupa* 'silent'.

3. **Differences in Character-Sets** : Due to differences in both these scripts, a one-to-one mapping cannot be developed between them. There are many characters which have more than one or no mapping in the other script.

6.3 Transliterating Sindhi Devanagari to Perso-Arabic

6.3.1 The Approach

Considering the challenges that we discussed earlier, we decided to go ahead with a rule-based transliteration technique since a large parallel text was not available to extract transliteration and apply machine learning techniques. The advantage of using rule-based approach is that it only needs a mapping between the characters in both writing forms. Then a system which converts characters in one script to another using this mapping or rules can be easily developed. Developing exhaustive set of rules is not possible in our case because the scripts do not have one-to-one mapping. There are some characters in Devenagari which have one-to-many or one-to-none mapping in Perso-Arabic.

We developed a character map (see complete mapping in Appendix Table B.1) and initiated our experiments. We handled the one-to-many cases by choosing the most frequently used character in Perso-Arabic. The system was evaluated on a test data of 842 parallel words obtained from 2 stories written in both the scripts by Mr Bhagwan Babani, a renowned Sindhi writer from India. This experiment gave us a word accuracy of 75%.

Devanagari	Perso-Arabic
इE	ءِ
Bइ	ا
अE	ءَ
उE	ءُ
Bए	اي
Bह	ھ
Bन्हE	نھ
Bभ्हE	ڀھ

BतE	تھ
BबिE	بھ
BबE	بھ
पुरE	پور
ऐं	ءٕ
में	ٕمِ

Table 6.2: Sindhi Devanagari to Perso-Arabic Mapping for character-combinations.

We analysed the output of this experiment and noticed interesting patterns of character combinations. We found that some words in Devanagari, such as में and ऐं always transliterate to a single character in Perso-Arabic. Similarly, we also noticed that some characters do not transliterate to most frequent mapping when they occur in certain contexts such beginning, end or middle of word. Therefore, we added these words and combination of characters (shown in Table 6.2) in our set of rules, with higher priority in order of transliteration and experimented with them. This experiment resulted an increase in accuracy to 82.56%.

6.3.2 Post-Processing for alternate spelling selection

In the error analysis of the previous experiment we found that there were many errors due to the one-to-many mapping of some characters (shown in Table 6.3). So far our output was the most frequent character among the many output alternatives for an input character. To solve this problem, a post-processing step was created which would process the output and check for its alternate spellings with different character options and finally output the best spelling.

We extracted a list of words which contained Perso-Arabic characters (shown in Table 6.3) from a large corpus of Perso-Arabic Sindhi (discussed in the next chapter). We also cleaned this list and normalised words containing diacritics to without diacritics. Then we created a dictionary out of it, where the key would be the word with most frequent characters (that our system shall output) and values would be a list of its alternate spellings obtained from the list. When the output from our system would match one of the keys, we would choose the best spelling among all the options based on the probablity of that spelling in Sindhi Perso-Arabic language model (created using the aforementioned large corpus). After incorporating this step we were able to create the best system so far and achieve an accuracy of 91.33%.

Perso-Arabic	**Devanagari**	**Perso-Arabic**	**Devanagari**
س	स	ت	त
ص	स	ط	त

ث	स	ح	ह
ز	ज़	ھ	ह
ذ	ज़	ق	फ़
ض	ज़	ظ	ज़

Table 6.3: One-to-Many Mappings between Sindhi Devanagari and Sindhi Perso-Arabic

6.3.3 Error Analysis

We realised that not only both Perso-Arabic and Devanagari scripts have orthographic differences but there are also differences in the way in which people write some particular words using either script. For instance, words with double consonants are written using 2 characters in Devanagari but only 1 in Perso-Arabic. Some examples are point 1 and 2 of Table 6.4.

There are cases where the pronunciation of a word is different than the written form. One such case is where the written form has long-vowel while the pronunciation has corresponding short-vowel. Some examples os such words are shown in points 3, 4 and 5 of Table 6.4. Another such case is where the Perso-Arabic word end with an extra ھ (see point 12 of Table 6.4). We also found some foreign words (see examples 6, 7 and 8 in Table 6.4) in our test data. Our transliterator has converted them into a phonetic equivalent but their written differs that the expected way of writing in Perso-Arabic.

The Perso-Arabic character ع and ء does not have an equivalent mapping in Devanagari and therefore causes lot of errors (see example 9-11 of Table 6.4).

S. No.	Output	Expectation	Input
1	سللي	صلي	सिल्ली
2	ابباس	عباس	अब्बास
3	سرشٹ	سرشتي	सृष्टि
4	ڪوتا	ڪويتا	कविता
5	مجب	موجب	मुजिबि
6	انجنيار	انجنيئر	इंजिनीअर
7	ڊااُن	ڊائون	डाउन
8	سپين	اسپين	स्पेन
9	شروآت	شروعات	शुरूआत
10	اُسمان	عثمان	उस्मान
11	ارسو	عرصو	अर्सो
12	ڪجھ	ڪجھھ	कुझु

Table 6.4: Types of errors in Sindhi Devanagari to Perso-Arabic Transliteration.

6.4 Part-of-Speech Tagger for Sindhi Perso-Arabic

Our motivation behind developing the transliteration system was to leverage Sindhi Devanagari, which had less digital resources, by converting it to Sindhi Perso-Arabic, which is a more resourceful script. Therefore, we used our transliteration system to convert our POS tagged corpus from Sindhi Devanagari to Sindhi Perso-Arabic and train a POS tagger. We trained a POS tagger in the same manner as Sindhi Devanagari. The training data had 4482 sentences and test data had 500 sentences. Our model obtained an accuracy of 92.28%.

6.4.1 Error Analysis

The accuracy of Sindhi Perso-Arabic tagger is a little lower than Sindhi Devanagari tagger. This was expected by us as there was an intermediate transliteration system involved, which is not perfect and therefore must have caused certain error propagation.

During our analysis we found similar patterns to that of Sindhi Devanagari tagger. For instance, proper-noun (NNP) and noun (NN) , verb (VM) and auxiliary verb (VAUX), adjective (JJ) and noun (NN), were some of the most hard to disambiguate pairs.

There was also another pair which was unusually high number of instances in among the errors. This was demonstrative (DMD) and personal-pronoun (PRP). An example of this was هن, which in Devanagari can be written as हिन (demonstrative) or हुन (personal-pronoun). Since we do not use diacritics in Perso-Arabic, these words of different categories became hard to disambiguate.

6.5 Transliterating Sindhi Perso-Arabic to Devanagari

This is direction of transliteration is very important because there is a large amount content available in Sindhi Perso-Arabic and it shall become accessible to the audience familiar with only Devanagari script.

6.5.1 The Approach

We started by adapting our previously developed character mapping (Table B.1). We adaptated including all the mappings except some of the character combination patterns and vowels. Using this mapping gave a drastically result with 8.67% accuracy. Then, based on errors analysis we included some combination patterns and reached an accuracy of 13.3%. Based on these result we concluded that rule-based approach is not suitable for transliterating from Sindhi Perso-Arabic to Devanagari script.

6.5.2 Error Analysis

The accuracy of our rule-based approach is very low due various reasons. One of the major reason is presence of dependent vowels (matras) in Devanagari script. The diacritics which could transliterate to these dependent vowels are usually missing in Perso-Arabic texts. Some examples for such errors are shown in points 1-4 of Table 6.5.

Another reason is that some characters in Sindhi Perso-Arabic, such as ي , ن , و have one-to-many mappings, across categories. For instance, و can be one of the vowels ौ, ो, ू or the consonant व. Examples of errors caused due to this are shown in Table 6.5 at points 5-10.

We could not employ a post-processor here because of two major reasons. First, the transliteration options (alternate spellings) generated shall consist of valid words in the language (example, سهنا -> सहना (to tolerate) or सुहिना (beautiful)) unlike the previous case (Devanagari to Perso-Arabic) where the post-processor was primarily meant for spell correction. Another reason is that we do not have a sufficiently large corpora in Sindhi Devanagari to create a reliable language model.

The other errors include absence of half-consonant (halant) marker (see examples 11-12 in Table 6.5). Some more errors were also described previously in Section 6.3.3.

S. No.	Input	Output	Expectation
1	ٻاهر	ब्राहर	ब्राहिर
2	اهڙن	अहड़न	अहिड़नि
3	آواز	आवाज़	आवाज़ु
4	مسلم	मसलम	मुस्लिमु
5	اوم	अवम	ओम
6	جادو	जादो	जादू
7	گوورڌن	गववरधन	गोवर्धन
8	انگلينڊ	अनगलयनड	इंगलैंड
9	اپگرهي	अपगरहय	अपगरही

10	اڳتي	अग्रतय	अग्रिते
11	ڪارل	कारल	कार्ल
12	بلٽ	बलट	बिल्ट

Table 6.5: Types of errors in Sindhi Perso-Arabic to Devanagari Transliteration.

S. No.	Input	Sangam	Expectation
1	उचो	اچو	اُچو
2	अशोक	اَشوڪ	اشوڪ
3	इंगलैंड	اِنگلينڊ	انگلينڊ
4	टालपुर	ٽالپر	ٽالپور
5	तस्वीरूं	تسويرون	تصويرون
6	हलु	هل	حل
7	ज़ख़ीरो	زخيرو	ذخيرو
8	ग़ाल्हि	ڳالھ	ڳالھھ
9	अबदाललतीफ़	ابداللتيف	عبداللطيف

Table 6.6: Comparison with Sangam for Perso-Arabic to Devanagari Sindhi transliteration.

6.5.3 Comparison with Sangam

Sangam is transliteration system created by Lehal et al. [47] for transliterating between Perso-Arabic scripts (Urdu , Shahmukhi and Sindhi) and Indic scripts (Hindi (Devanagari), Gurmukhi and Sindhi (Devanagari)). This system is a hybrid system which is created using, linguistic rules, character and word language models and large number transliteration pairs. The system reports 91.68% word accuracy for transliteration from Perso-Arabic to Devanagari Sindhi, while accuracy for the reverse direction is not reported.

We evaluated their system using our test set and obtained 90.5% word accuracy for Devanagari to Perso-Arabic Sindhi transliteration and 93% for the reverse direction. Therefore, the performance of rule-based approach is almost at-par with hybrid approach for Sindhi Devanagari to Perso-Arabic transliteration and hybrid approach is much better for the other way.

After analysing the output of Sangam on our evaluation set. We found the following differances and similarities between the two systems while transliterating from Devanagari to Perso-Arabic Sindhi. We shall discuss them below and illustrate the examples in Table 6.6.

1. The Perso-Arabic output of our system is without daicritics, with a few exceptions. Similarly, Sangam also outputs daicritics inconcistently (example 1-3).

2. Character combinations provided us an edge in some cases, ideally these should have been covered by character LM in Sangam (example 4).

3. Some of the errors in output of Sangam were also due to incorrect spellings of words with one-to-many mapping letters (example 5-7).

4. Both the systems fails in special words with orthographic and phonetic dissimilarities or where letters do not have an exact mapping (examples 8-10).

6.6 Leveraging Sindhi Devanagari POS Tagger

We had discussed previously that Sindhi Devanagari is low-resourced. In this section we shall demonstrate our experiments in enriching Sindhi Devanagari resources by leveraging Sindhi Perso-Arabic data and transliteration system. We shall generate unlabeled data in Sindhi Devanagari by transliterating data from Sindhi Perso-Arabic. This shall further be used in generating more training data for POS tagger though bootstrapping.

6.6.1 Experimental Setup

We collected a set of 500 sentences in Sindhi Perso-Arabic. Then, we transliterated them through Sangam [47] into Sindhi Devanagari. After some cleaning and normalization we had 493 unlabeled sentences.

We ran POS tagger on these sentences to obtained their POS labels. This labeled data was fed into the training data and POS tagger was re-trained and re-tested. We experimented by trainind 3 POS taggers with different features. The first is CRF-1, it consists of only consists only contextual features. The second is CRF-4, which also has morphological and lexical features. The third is CRF-6, this consists of all the previous features alongwith categorical features. The details of these models had been discussed in Section 4.3. The results and error analysis of the experiments have been discussed below.

6.6.2 Error Analysis

We observed that all the bootstrapped taggers had lower accuracy than before (see Table 6.7). This behavior was expected as the pipeline had a transliteration module which is not perfect and therefore errors were propagated in the POS tagging module as well. Although, decline in accuracy was expected but its margin was not very high. Considering Sindhi Perso-Arabic also

	CRF-1	CRF-4	CRF-6
Before	87.80%	92.41%	92.69%
After	87.04%	91.78%	92.43%

Table 6.7: Accuracy of different POS tagging models before and after bootstraping.

has Urdu and Arabic influence, the decline could have been much higher due to large number of unknowns. The accuracy of CRF-1 and CRF-4 fell consistently by $\tilde{0}$.8% and interestingly the accuracy of CRF-6 fell by lesser margin of $\tilde{0}$.4%. This also shows adding categorical features are supporting better tagging Quantifiers and Verbs in unlabeled data.

We also conducted some more experiments with CRF-6 and using limited amount of unlabeled data for bootstrapping, which was gradually increased. We saw that the accuracy dropped linearly with increase in size of bootstrapped training data (see Table 6.8). The reason for drop in accuracy is that bootstraped data already consists of errors that model does initially and we are retraining without correction of these annotations which makes the bootstapped model learn errorneously.

Sentences	**100**	**300**	**493**
Accuracy	92.54%	92.48%	92.43%

Table 6.8: POS tagger accuracy on increasing bootstrapped data.

6.7 Conclusion

Transliteration is a great tool to reduce the script barrier between Devanagari and Perso-Arabic scripts of Sindhi language. We built a rule-based transliteration system converting Sindhi Devanagari to Perso-Arabic with an accuracy of 91.33%. Similarly, we conducted rule-based transliteration for the reverse direction and the low-accuracy numbers show that rule-based is not the right approach for Perso-Arabic to Devanagari transliteration. We also demonstrated how resources can be leveraged among these scripts by using a transliteration system. We built a POS tagger for Sindhi Perso-Arabic which gives an accuracy of 92.28%. This is was built using our POS annotated data in Sindhi Devanagari. Similarly, we also created more training data for Sindhi Devanagari POS tagger by getting data from Sindhi Perso-Arabic. The bootstrapped POS tagger is usable as it gives only 0.4% less accuracy than the best tagger.

Chapter *7*

Conclusions and Future Directions

Sindhi is an Indo-Aryan language with large community of speakers worldwide. It is also an official language in India and Pakistan. Despite the large community Sindhi is a computationally resource poor language. The aim of this research was to empower Sindhi language for natural language processing by developing various fundamental tools and resources that are important for a language in computational linguistics.

We have built raw and part-of-speech dataset for Sindhi Devanagari. This was also used to train an automatic POS tagger using Conditional Random Fields that performs with 91.78% average accuracy. In future this work can be extended to create a better POS tagger by using more annotated data and various techniques like Deep Learning. This can also be in constructing Shallow parsers and Dependancy parsers for Sindhi.

We have also built a paradigm based finite-state morphological analyser for Sindhi Perso-Arabic using Apertium's lttoolbox. This morphological analyser currently has about 3500 entries and a coverage of more than 81% on Sindhi Wikipedia consisting of 341.5k tokens. Apart from improving the analyser, this work can extended to various applications in future. The morphological analyser presented in this work also has a monolingual dictionary which is an essential part of a language pair in rule-based machine translation systems in Apertium. We already have an Urdu-Hindi MT system in Apertium and since these languages are closely related and share common linguistic properties, we can use them to develop MT systems for Urdu-Sindhi and Sindhi-Hindi language pairs.

Transliteration is a great tool to reduce the script barrier between Devanagari and Perso-Arabic scripts of Sindhi language. We built a rule-based transliteration system converting Sindhi Devanagari to Perso-Arabic with an accuracy of 91.33%. In future, this work can be used to leverage various resources developed in either scripts. We had demonstrated in our work how more data for Sindhi Devanagari can be created and how experimented with a transliterated

POS tagger for Sindhi Perso-Arabic. Similarly, the morph features could be used to improve the Devanagari POS tagger. Both these modules can also be combined to create a full parser. Transliterator could also be used to create script independant applications and tools.

7.1 Summary of Observations on Sindhi

This section aims to list down all the observations made about and challenges faced with Sindhi during the course of various experiments and resource developments. These properties can be exploited by researchers working on Sindhi and shall help them in using or improving the existing tools.

1. Role of Daicritics - Daicritics play a very important role in the ambiguity of Perso-Arabic texts. Diacritic restoration for Perso-Arabic Sindhi would greatly and positively impact the performance of all tools made for Sindhi Perso-Arabic

2. Role of Morphology - In POS Tagging for Devanagari, we saw the including 3 prefix and 6 suffixes gained a lot of accuracy. Hence, Sindhi's morphology may be covered in the last 6 characters of a word. Also, a proper morpho-analyser in Sindhi Devanagari might positively impact POS tagging performance due to better feature selection.

3. Role of Size of Training Data - In POS tagging, we observed the curve of accuracy has not yet reached a plateau and we shall definitely benefit from more training data. More training data can be created or bootstrapping can be explored for POS Tagging in Sindhi Devanagari.

4. Lexical Ambiguity - In POS tagging, we have observed noun-adjective and Verb-auxiliary verb ambiguity. Tools such as NER (Named-Entity recognition) might help with the noun ambiguity.

5. Orthographic Normalization - Languages written using Perso-Arabic scripts have a lot of common characters and Unicode interjections. A Unicode Normalizer for such languages can help in creation of cleaned corpus and subsequently better NLP tools.

6. Leveraging Devanagari - Sindhi Devanagari is more resource-poor as compared to Sindhi Perso-Arabic and transliteration in this direction is very hard as well. Although, we conducted some experiments with existing transliteration to show results on leveraging Devanagari. More such experiments can be conducted or work can be done on improvement of transliteration.

Related Publications

1. **Title**: Developing Part-of-Speech Tagger for a Resource Poor Language: Sindhi
   Authors: Raveesh Motlani, Harsh Lalwani, Manish Shrivastava and Dipti M. Sharma
   Published: In Proceedings of the 7th Language and Technology Conference (LTC 2015)

2. **Title**: A finite-state morphological analyser for sindhi
   Authors: Raveesh Motlani, Francis M. Tyers and Dipti M. Sharma
   Published: In Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)

3. **Title**: Developing language technology tools and resources for a resource-poor language: Sindhi
   Author: Raveesh Motlani
   Published: In Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT, 2016)

Appendix *A*

BIS Parts-Of-Speech Tagset for Sindhi

Sl. No	Category		Label	Annotation Convention
	Top level	Subtype(level 1)		
1	**Noun**		N	N
1.1		Common	NN	N_NN
1.2		Proper	NNP	N_NNP
1.3		Spatio-Temporal	NST	N_NST
2	**Pronoun**		PR	PR
2.1		Personal	PRP	PR_PRP
2.2		Reflexive	PRF	PR_PRF
2.3		Relative	PRL	PR_PRL
2.4		Reciprocal	PRC	PR_PRC
2.5		Wh-word	PRQ	PR_PRQ
2.6		Indefinite	PRI	PR_PRI
3	**Demonstrative**		DM	DM
3.1		Deictic	DMD	DM_DMD
3.2		Relative	DMR	DM_DMR
3.3		Wh-word	DMQ	DM_DMQ
3.4		Indefinite	DMI	DM_DMI
4	**Verb**		V	V
4.1		Main	VM	V_VM
4.2		Auxiliary	VAUX	V_VAUX

5	**Adjective**		JJ	
6	**Adverb**		RB	
7	**Postposition**		PSP	
8	**Conjunction**		CC	CC
8.1		Co-ordinator	CCD	CC_CCD
8.2		Subordinator	CCS	CC_CCS
9	**Particles**		RP	RP
9.1		Default	RPD	RP_RPD
9.2		Classifier	CL	RP_CL
9.3		Interjection	INJ	RP_INJ
9.4		Intensifier	INTF	RP_INTF
9.5		Negation	NEG	RP_NEG
10	**Quantifiers**		QT	QT
10.1		General	QTF	QT_QTF
10.2		Cardinals	QTC	QT_QTC
10.3		Ordinals	QTO	QT_QTO
11	**Residuals**		RD	RD
11.1		Foreign word	RDF	RD_RDF
11.2		Symbol	SYM	RD_SYM
11.3		Punctuation	PUNC	RD_PUNC
11.4		Unknown	UNK	RD_UNK
11.5		Echowords	ECH	RD_ECH

Table A.1: Adapted BIS Tagset for Sindhi

Appendix *B*

Character Mapping between Sindhi Devanagari and Sindhi Perso-Arabic

Devanagari	Perso-Arabic
Character Combinations	
इE	ءِ
Bइ	ا
अE	ءَ
उE	ءُ
Bए	اي
Bह	ھ
BनE	نھ
Bब्रE	ٻھ
BतE	تھ
BबिE	بھ
BबE	بھ
पुरE	پور
ऐं	ءٔ
में	۾
Consonants	
ड़	ڙ
ढ़	ڙھ
क़	ق
ख़	خ

ه	ह
ي	य
و	व
ڻ	ण
ن	न
م	म
ل	ल
گ	ग
ک	ख
ڪ	क
ف	फ
ش	ष
ش	श
ر	र
ڍ	ढ
ڈ	ड
د	द
ڌ	ध
ڇ	छ
چ	च
ج	ञ
پ	ज
ٺ	प
ٽ	ठ
ٹ	ट
ٿ	थ
ت	त
ٻ	भ
ب	ब
ڀ	झ
ڦ	घ
ف	फ़
ز	ज़
گ	ङ
ڳ	ग़

स	س
Sindhi Implosives	
ब॒	ٻ
ड॒	ڏ
ज॒	ڄ
ग॒	ڳ
Vowels	
आ	آ
अ	ا
ा	ا
ौ	و
ू	و
ी	ي
ो	و
ं	ن
ृ	ر
औ	عو
इ	ع
ई	ئي
ऐ	اع
ए	ئي
ऊ	ئو
उ	اُ
ओ	او
ै	ي
ै॒	ي
ऑ	آ

Table B.1: Character Mapping between Sindhi Devanagari and Sindhi Perso-Arabic

Bibliography

[1] N. AbdulJaleel and L. S. Larkey. Statistical transliteration for english-arabic cross language information retrieval. In *Proceedings of the twelfth international conference on Information and knowledge management*, pages 139–146. ACM, 2003.

[2] M. Arbabi, S. M. Fischthal, V. C. Cheng, and E. Bart. Algorithms for arabic name transliteration. *IBM Journal of research and Development*, 38(2):183–194, 1994.

[3] P. Arulmozhi and L. Sobha. A hybrid pos tagger for a relatively free word order language. In *Proceedings of the First National Symposium on Modeling and Shallow Parsing of Indian Languages*, pages 79–85, 2006.

[4] R. E. Banchs, M. Zhang, X. Duan, H. Li, and A. Kumaran. Report of news 2015 machine transliteration shared task. In *Proceedings of NEWS 2015 The Fifth Named Entities Workshop*, page 10, 2015.

[5] M. Bapat, H. Gune, and P. Bhattacharyya. A paradigm-based finite state morphological analyzer for marathi. In *Proceedings of the 1st Workshop on South and Southeast Asian Natural Language Processing (WSSANLP)*, pages 26–34, 2010.

[6] A. Bharati, V. Chaitanya, R. Sangal, and K. Ramakrishnamacharyulu. *Natural language processing: a Paninian perspective*. Prentice-Hall of India New Delhi, 1995.

[7] E. Black, F. Jelinek, J. Lafferty, R. Mercer, and S. Roukos. Decision tree models applied to the labeling of text with parts-of-speech. *In Proc. of Workshop on Speech and Natural Language (HLT/ACL)*, page 117–121, 1992.

[8] T. Brants. Tnt: A statistical part-of-speech tagger. *Proceedings of the Sixth Conference on Applied Natural Language Processing*, pages 224–231, 2000.

[9] S. Chaware and S. Rao. Rule-based phonetic matching approach for hindi and marathi. *International Journal of Research in Social Sciences*, 1(1):26, 2011.

[10] J. Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, page 37–46, 1960.

[11] M. Creutz and K. Lagus. *Unsupervised morpheme segmentation and morphology induction from text corpora using Morfessor 1.0*. Helsinki University of Technology, 2005.

[12] D. Cutting, J. Kupiec, J. Pedersen, and P. Sibun. A practical part-of-speech tagger. *In Proc. of 3rd Conference on Applied Natural Language Processing*, 1992.

[13] C. Daswani. Movement for the recognition of sindhi and choice of a script for sindhi. *Language Movements in India*, pages 60–69, 1979.

[14] D. Demner-Fushman and D. W. Oard. The effect of bilingual term list size on dictionary-based cross-language information retrieval. In *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*, pages 10–pp. IEEE, 2003.

[15] V. Dhanalakshmi, R. Rekha, A. Kumar, K. Soman, S. Rajendran, et al. Morphological analyzer for agglutinative languages using machine learning approaches. In *Advances in Recent Technologies in Communication and Computing, 2009. ARTCom'09. International Conference on*, pages 433–435. IEEE, 2009.

[16] X. Duan, R. E. Banchs, M. Zhang, H. Li, and A. Kumaran. Report of news 2016 machine transliteration shared task. *ACL 2016*, page 58, 2016.

[17] N. Durrani, H. Sajjad, A. Fraser, and H. Schmid. Hindi-to-urdu machine translation through transliteration. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 465–474. Association for Computational Linguistics, 2010.

[18] N. Durrani, H. Sajjad, H. Hoang, and P. Koehn. Integrating an unsupervised transliteration model into statistical machine translation. In *EACL*, volume 14, pages 148–153, 2014.

[19] A. Ekbal, R. Haque, and S. Bandyopadhyay. Bengali part of speech tagging using conditional random field. *In Proc.of the 7th International Symposium on Natural Language Processing (SNLP-07)*, 2007.

[20] A. Finch and E. Sumita. Phrase-based machine transliteration. In *Proceedings of the Workshop on Technologies and Corpora for Asia-Pacific Speech Translation (TCAST)*, pages 13–18, 2008.

[21] M. L. Forcada, M. Ginestí-Rosell, J. Nordfalk, J. O'Regan, S. Ortiz-Rojas, J. A. Pérez-Ortiz, F. Sánchez-Martínez, G. Ramírez-Sánchez, and F. M. Tyers. Apertium: a free/open-source platform for rule-based machine translation. *Machine Translation*, 25(2):127–144, 2011.

[22] M. Frodl. *Part-of-Speech Tagging Using Neural Networks*. PhD thesis, Doctoral dissertation, Masarykova univerzita, Fakulta informatiky, 2014.

[23] M. Ganapathiraju, M. Balakrishnan, N. Balakrishnan, and R. Reddy. Om: One tool for many (indian) languages. *JOURNAL-ZHEJIANG UNIVERSITY SCIENCE*, 6(11):1348, 2005.

[24] R. Garside and N. Smith. A hybrid grammatical tagger: Claws4. *Corpus annotation: Linguistic information from computer text corpora*, pages 102–121, 1997.

[25] J. Goldsmith. Unsupervised learning of the morphology of a natural language. *Comput. Linguist.*, pages 153–198, 2001.

[26] V. Goyal and G. S. Lehal. Hindi morphological analyzer and generator. In *Emerging Trends in Engineering and Technology, 2008. ICETET'08. First International Conference on*, pages 1156–1159. IEEE, 2008.

[27] V. Goyal and G. S. Lehal. Hindi-punjabi machine transliteration system (for machine translation system). *George Ronchi Foundation Journal, Italy*, 64(1):2009, 2009.

[28] B. B. Greene and G. M. Rubin. *Automatic grammatical tagging of English*. Department of Linguistics, Brown University, 1971.

[29] K. Gupta, M. Choudhury, and K. Bali. Mining hindi-english transliteration pairs from online hindi lyrics. In *LREC*, pages 2459–2465, 2012.

[30] R. Gupta, P. Goyal, and S. Diwakar. Transliteration among indian languages using wx notation. In *KONVENS*, pages 147–150, 2010.

[31] L. Haizhou, Z. Min, and S. Jian. A joint source-channel model for machine transliteration. In *Proceedings of the 42nd Annual Meeting on association for Computational Linguistics*, page 159. Association for Computational Linguistics, 2004.

[32] Z. S. Harris. *String analysis of sentence structure*. Number 1. Mouton, 1962.

[33] U. Hermjakob, K. Knight, and H. Daumé III. Name translation in statistical machine translation-learning when to transliterate. In *ACL*, pages 389–397, 2008.

[34] S. M. Idicula and P. S. David. A morphological processor for malayalam language. *South Asia Research*, 27(2):173–186, 2007.

[35] A. Irvine, C. Callison-Burch, and A. Klementiev. Transliterating from all languages. In *Proceedings of the Conference of the Association for Machine Translation in the Americas (AMTA)*, pages 100–110, 2010.

[36] A. Jalal. *The sole spokesman: Jinnah, the Muslim League and the demand for Pakistan*, volume 31. Cambridge University Press, 1994.

[37] I. Jena, S. Chaudhury, H. Chaudhry, and D. M. Sharma. Developing oriya morphological analyzer using lt-toolbox. In *Information Systems for Indian Languages*, pages 124–129. Springer, 2011.

[38] N. Kanuparthi, A. Inumella, and D. M. Sharma. Hindi derivational morphological analyzer. In *Proceedings of the Twelfth Meeting of the Special Interest Group on Computational Morphology and Phonology*, pages 10–16. Association for Computational Linguistics, 2012.

[39] F. Karlsson, A. Voutilainen, J. Heikkilae, and A. Anttila. *Constraint Grammar: a language-independent system for parsing unrestricted text*, volume 4. Walter de Gruyter, 1995.

[40] S. Klein and R. F. Simmons. A computational approach to grammatical coding of english words. *Journal of the ACM (JACM)*, 10(3):334–347, 1963.

[41] A. Klementiev and D. Roth. Weakly supervised named entity transliteration and discovery from multilingual comparable corpora. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 817–824. Association for Computational Linguistics, 2006.

[42] A. Kumaran, M. M. Khapra, and H. Li. Report of news 2010 transliteration mining shared task. In *Proceedings of the 2010 Named Entities Workshop*, pages 21–28. Association for Computational Linguistics, 2010.

[43] A. Kunchukuttan, R. Puduppully, and P. Bhattacharyya. Brahmi-net: A transliteration and script conversion system for languages of the indian subcontinent. In *HLT-NAACL*, pages 81–85, 2015.

[44] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *In Proceedings of ICML*, 2001.

[45] M. Leghari and M. U. Rahman. Towards transliteration between sindhi scripts by using roman script. In *Conference on Language and Technology*, 2010.

[46] G. S. Lehal and T. S. Saini. A hindi to urdu transliteration system. In *Proceedings of ICON-2010: 8th International Conference on Natural Language Processing, Kharagpur*, 2010.

[47] G. S. Lehal and T. S. Saini. Sangam: A perso-arabic to indic script machine transliteration model. In *11th International Conference on Natural Language Processing*, page 232, 2014.

[48] H. Li, A. Kumaran, V. Pervouchine, and M. Zhang. Report of news 2009 machine transliteration shared task. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*, pages 1–18. Association for Computational Linguistics, 2009.

[49] H. Li, A. Kumaran, M. Zhang, and V. Pervouchine. Report of news 2010 transliteration generation shared task. In *Proceedings of the 2010 Named Entities Workshop*, pages 1–11. Association for Computational Linguistics, 2010.

[50] J. A. Mahar and G. Q. Memon. Rule based part of speech tagging of sindhi language. *International Conference on Signal Acquisition and Processing*, pages 101–106, 2010.

[51] J. A. Mahar and G. Q. Memon. Sindhi part of speech tagging system using wordnet. *International Journal of Computer Theory and Engineering*, 2(4):538, 2010.

[52] J. A. Mahar and G. Q. Memon. Lexicon based diacritic restorations using wordnet for sindhi. *International Journal of Academic Research*, 3(2), 2011.

[53] J. A. Mahar and G. Q. Memon. Probabilistic analysis of sindhi word prediction using n-grams. *Australian Journal of Basic and Applied Sciences*, 5(5):1137–1143, 2011.

[54] J. A. Mahar, G. Q. Memon, and S. H. Danwar. Algorithms for sindhi word segmentation using lexicon-driven approach. *International journal of academic research*, 3(3), 2011.

[55] J. A. Mahar, G. Q. Memon, and S. H. A. Shah. Wordnet based sindhi text to speech synthesis system. In *Computer Research and Development, 2010 Second International Conference on*, pages 20–24. IEEE, 2010.

[56] M. G. Malik. Punjabi machine transliteration. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 1137–1144. Association for Computational Linguistics, 2006.

[57] D. K. Malladi and P. Mannem. Statistical morphological analyzer for hindi. In *IJCNLP*, pages 1007–1011, 2013.

[58] S. Mohanty, P. K. Santi, and K. D. Adhikary. Analysis and design of oriya morphological analyser: Some tests with orinet. In *Proceeding of symposium on Indian Morphology, phonology and Language Engineering, IIT Kharagpur*, 2004.

[59] P. Nainwani. Blurring the demarcation between machine assisted translation (mat) and machine translation (mt): the case of english and sindhi. In *Workshop on Indian Language and Data: Resources and Evaluation Workshop Programme*, page 102, 2012.

[60] P. Nainwani. Handling conflational divergence in a pair of languages: the case of english and sindhi. In *Workshop on Indian Language and Data: Resources and Evaluation Workshop Programme*, page 56, 2012.

[61] P. Nainwani. *Challenges in Automatic Translation of Natural Languages: A Case of English-Sindhi Divergence [with specific reference to Conflational divergence]*. Doctoral dissertation, Jawaharlal Nehru University, New Delhi, 2015.

[62] P. Nakov and J. Tiedemann. Combining word-level and character-level models for machine translation between closely-related languages. In *Proceedings of the 50th Annual Meeting of the As-*

sociation for Computational Linguistics: Short Papers-Volume 2, pages 301–305. Association for Computational Linguistics, 2012.

[63] K. Nongmeikapam and S. Bandyopadhyay. A transliteration of crf based manipuri pos tagging. *In Proc. of 2nd International Conference on Communication, Computing & Security*, page 582–589, 2012.

[64] J. D. Oad. *Implementing GF Resource Grammar for Sindhi language*. Msc. thesis, Chalmers University of Technology, Gothenburg, Sweden, 2012.

[65] F. J. Och and H. Ney. A systematic comparison of various statistical alignment models. *Computational linguistics*, 29(1):19–51, 2003.

[66] J.-H. Oh and K.-S. Choi. An english-korean transliteration model using pronunciation and contextual rules. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics, 2002.

[67] K. Parameswari. An improvized morphological analyzer cum generator for tamil: A case of implementing the open source platform apertium. *Knowledge Sharing Event*, 2010.

[68] C. Patel and K. Gali. Part-of-speech tagging for gujarati using conditional random fields. *Proceedings of the IJCNLP*, page 117–122, 2008.

[69] K. Patel and J. Pareek. Gh-map-rule based token mapping for translation between sibling language pair: Gujarati-hidi. In *Proceedings of International Conference on Natural Language Processing*, 2009.

[70] D. Pinto, A. McCallum, X. Wei, and W. B. Croft. Table extraction using conditional random fields. *In Proc. of the ACM SIGIR*, 2003.

[71] M. Rahman and S. K. Sarma. An implementation of apertium based assamese morphological analyzer. *arXiv preprint arXiv:1503.03989*, 2015.

[72] M. U. Rahman. Towards sindhi corpus construction. *Conference on Language and Technology, Lahore, Pakistan*, 2010.

[73] M. U. Rahman and M. I. Bhatti. Finite state morphology and sindhi noun inflections. *In Proceedings of the 24th Pacific Asia Conference on Language, Information and Computation, PACLIC 24, Tohoku University, Japan*, pages 669–676, 2010.

[74] T. Rama and K. Gali. Modeling machine transliteration as a phrase based statistical machine translation problem. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*, pages 124–127. Association for Computational Linguistics, 2009.

[75] P. Ranjan, H. V, A. Basu, and S. Sarkar. Part of speech tagging and local word grouping techniques for natural language parsing in hindi. In *Proceedings of the 1st International Conference on Natural Language Processing (ICON 2003)*, 2003.

[76] A. Ranta. Gf: A multilingual grammar formalism. *Language and Linguistics Compass*, 3, 2009.

[77] A. Ratnaparkhi. A maximum entropy model for part-of-speech tagging. *In Proc. of EMNLP*, pages 133–142, 1996.

[78] O. Rinju, R. Rajeev, and E. Sherly. Morphological analyzer for malayalam: Probabilistic method vs rule based method. 2013.

[79] F. Sha and F. Pereira. Shallow parsing with conditional random fields. *In Proc. of the NAACL-HLT, Canada*, page 134–141, 2003.

[80] B. R. Shambhavi and R. Kumar P. Kannada part-of-speech tagging with probablistic classifiers. *International Journal of Computer Applications*, 2012.

[81] T. Sherif and G. Kondrak. Bootstrapping a stochastic transducer for arabic-english transliteration extraction. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 45, page 864, 2007.

[82] M. Shrivastava, R. Melz, S. Singh, K. Gupta, and P. Bhattacharya. Conditional random field based pos tagger for hindi. *In Proc. of the MSPIL, Bombay*, pages 63–68, 2006.

[83] R. Srivastava and R. A. Bhat. Transliteration systems across indian languages using parallel corpora. In *PACLIC*, 2013.

[84] R. Udupa, K. Saravanan, A. Bakalov, and A. Bhole. "they are out there, if you know where to look": Mining transliterations of oov query terms for cross-language information retrieval. In *European Conference on Information Retrieval*, pages 437–448. Springer, 2009.

[85] A. Van den Bosch and W. Daelemans. Memory-based morphological analysis. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 285–292. Association for Computational Linguistics, 1999.

[86] T. Vikram and S. R. Urs. Development of prototype morphological analyzer for he south indian language of kannada. In *International Conference on Asian Digital Libraries*, pages 109–116. Springer, 2007.

[87] P. Vinod, V. Jayan, V. Bhadran, and C. Thiruvananthapuram. Implementation of malayalam morphological analyzer based on hybrid approach. *ROCLING XXIV (2012)*, page 307, 2012.

[88] P. Virga and S. Khudanpur. Transliteration of proper names in cross-lingual information retrieval. In *Proceedings of the ACL 2003 workshop on Multilingual and mixed-language named entity recognition-Volume 15*, pages 57–64. Association for Computational Linguistics, 2003.

[89] R. Wicentowski. Multilingual noise-robust supervised morphological analysis using the word-frame model. In *Proceedings of the 7th Meeting of the ACL Special Interest Group in Computational Phonology: Current Themes in Computational Phonology and Morphology*, pages 70–77. Association for Computational Linguistics, 2004.

[90] D. Zelenko and C. Aone. Discriminative methods for transliteration. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 612–617. Association for Computational Linguistics, 2006.

[91] M. Zhang, H. Li, A. Kumaran, and M. Liu. Report of news 2011 machine transliteration shared task. Association for Computational Linguistics, 2009.

[92] M. Zhang, H. Li, A. Kumaran, and M. Liu. Report of news 2012 machine transliteration shared task. In *Proceedings of the 4th Named Entity Workshop*, pages 10–20. Association for Computational Linguistics, 2012.