A Tool for Transliteration of Bilingual Texts Involving Sanskrit

Nikhil Chaturvedi
IIT Delhi
`cs5130291@cse.iitd.ac.in`

Prof. Rahul Garg
IIT Delhi
`rahulgarg@cse.iitd.ac.in`

Abstract

Sanskrit texts are increasingly being written in bilingual and trilingual formats, with Sanskrit paragraphs/shlokas followed by their corresponding English commentary. Sanskrit can also be written in many ways, including multiple encodings like SLP-1 and Velthuis for its romanised form. The need to tackle such code-switching is exacerbated through the requirement to render web pages with multilingual Sanskrit content. These need to automatically detect whether a given text fragment is in Sanskrit, followed by the identification of the form/encoding, further selectively performing transliteration to a user specified script. The Brahmi-derived writing systems of Indian languages are mostly rather similar in structure, but have different letter shapes. These scripts are based on similar phonetic values which allows for easy transliteration. This correspondence forms the basis of the motivation behind deriving a uniform encoding schema that is based on the underlying phonetic value rather than the symbolic representation. The open-source tool developed by us performs this end-to-end detection and transliteration, and achieves an accuracy of 99.1% between SLP-1 and English on a Wikipedia corpus using simple machine learning techniques.

1 Introduction

Sanskrit is one of the most ancient languages in India and forms the basis of numerous Indian languages. It is the only known language which has a built-in scheme for pronunciation, word formation and grammar (Maheshwari, 2011). It one of the most used languages of it's time (Huet et al., 2009) and hence encompasses a rich tradition of poetry and drama as well as scientific, technical, philosophical and religious texts. Unfortunately, Sanskrit is now spoken by only a small number of people. The aforementioned literature, though available, remains inaccessible to most of the world. However, in recent years, Sanskrit has shown a resurgence through various media, with people reviving the language over the internet (Dasgupta, 2014) and through bilingual and trilingual texts.

There exist numerous web-based tools applications that provide age-old Sanskrit content to users and assist them with getting an insight into the language. Cologne Sanskrit Dictionary Project (Kapp and Malten, 1997) aims to digitize the major bilingual Sanskrit dictionaries. Sanskrit Reader Companion (Goyal and Huet, 2013) by Inria has tools for declension, conjugation, Sandhi splitting and merging along with word stemming. Samsadhani (Goyal et al., 2012) - A Sanskrit Computational Toolkit by University of Hyderabad. Sanskrit language processing tools developed at the Jawaharlal Nehru University (Jha et al., 2009). In this paper, we attempt to construct a tool to render the web pages of the above tools in multiple scripts and encodings at the backend, based on the requirements. Through this, not only do we aim to expand the reach of Sanskrit to a wider community, but also to standardize an open-source tool for transliteration.

The number of bilingual and trilingual textual material involving Sanskrit has also been on a steady rise. For example, Gita Supersite maintained by IIT Kanpur serves as a huge bilingual database of the

Bhagvad Gita, the Ramacharitmanas and Upanishads. There are also traditional texts which exist in a similar format like Srisa Chandra Vasu's translation of the Ashtadhyayi in English (Vasu, 1897). These works broadly follow a commentary structure with Sanskrit hyms, verses and words being followed by their translation in popular modern day languages like English or Hindi. Code-switching (Auer, 2013) is the practice of moving back and forth between two languages, or between two dialects/registers of the same language. Due to their commentarial nature, multilingual Sanskrit works constitute massive amounts of code-switching. For example, an excerpt of the Valmiki Ramayana from Gita Supersite: "तपस्वी ascetic, वाल्मीकि: Valmiki, तप: स्वाध्यायनिरतम् highly delighted in the practice of religious austerities and study of vedas, वाग्विदां वरम् eloquent among the knowledgeable, मुनिपुङ्गवम् preeminent among sages, नारदम् Narada, परिपप्रच्छ enquired." This motivates the need for a word-level transliteration tool that tackles areas of code-switching and performs transliteration through an automatic detection of the relevant sections.

Another interesting aspect that has led to the resurgence of Sanskrit has been the concept of Romanisation which, in linguistics, is the conversion of writing from a different writing system to the Roman (Latin) script. Multiple methods of this transliteration have emerged, although none has emerged as the clear standard. These standards include SLP1, Velthuis, Harvard-Kyoto, ISO15919, IAST and National Library at Kolkata romanization. Such romanisation makes it easy for large parts of the population to pronounce and appreciate Sanskrit verses. Hence, any standardized transliteration tool for Sanskrit needs to support all the above romanisation encodings.

An interesting property of Sanskrit and other major Indian languages like Hindi, Marathi, Tamil, Gujarati etc. forms the basis of our transliteration and auto-detection. These languages are written using different letter shapes (scripts) but are rather similar structurally as the same sounds are duplicated across these allowing for easy transliteration. The phonetic sound [ki] will be rendered as कि in Devanagari, as ਕਿ in Gurmukhi, and as கி in Tamil. Each having different code- points in Unicode and ISCII. This enabled us to formulate a mediating encoding schema that encodes the sound of a syllable rather than any syntactical aspect, thus allowing seamless transliteration between any 2 given scripts.

Including Romanised Sanskrit however exacerbates the problem of code-switching. The requirement is now to differentiate between two words of the same script, which turns out to be a non-trivial problem. We again use the intuition of phonetics to overcome this problem. Certain sounds (or chain of sounds) occur more frequently in some languages than in others. This allows us to formulate the classifier using a simple Naive Bayes model that functions on all possible substrings of a given word. We manage to achieve a classification accuracy of 99.1% between English and SLP1.

2 Sanskrit Alphabets and Encodings

The Sanskrit alphabet consists of 5 short vowels, 8 long vowels and 9 pluta vowels. Each of these vowels can be pronounced in three different ways: Udaatta (Acute accent, high pitch), Anudaatta (Grave accent, low pitch) and Svarita (Circumflex, high falling pitch). Vowels in udaatta mode are written as before, in anudaatta mode, a horizontal line is drawn under them and svarita vowels are written with a vertical line drawn above them. There are 33 consonants which includes 4 semi-vowels and 3 sibilants and 1 aspirate (ha).

There are several methods of transliteration from Devanagari to the Roman script (a process known as romanization) which share similarities, although no single system of transliteration has emerged as the standard. Eg. SLP1, Velthius, Harvard-Kyoto etc. These can represent not only the basic Devanagari letters, but also phonetic segments, phonetic features and punctuation. SLP1 also describes how to encode classical and Vedic Sanskrit. A comparison of these schemata is given in Table 2.

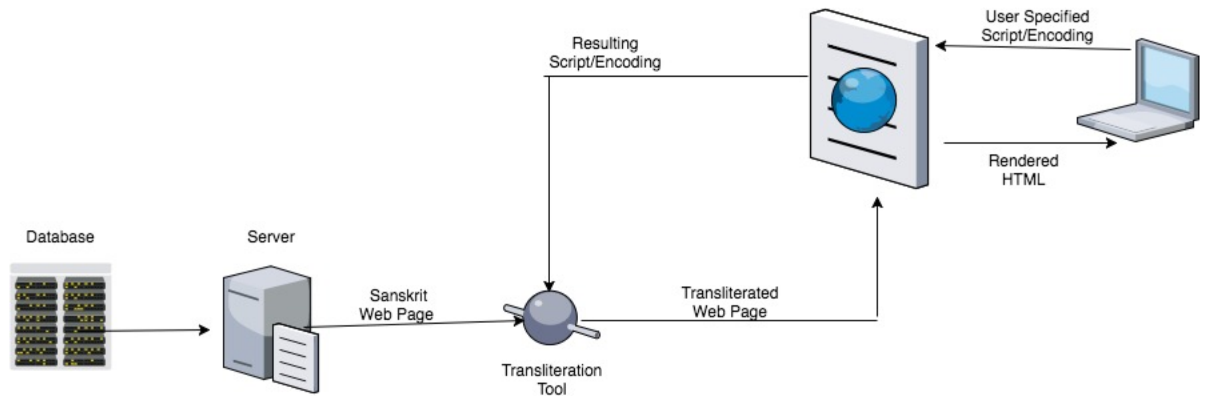


Figure 1: Model for Web-Based Applications

Unicode has designated code blocks for almost all major Indian scripts. The supported scripts are: Assamese, Bengali (Bangla), Devanagari, Gujarati, Gurmukhi, Kannada, Malayalam, Oriya, Tamil, and Telugu among others. Across scripts, Unicode respects alphabet correspondence and letters with similar phonetic values are assigned the same code-points. As a result, transliteration can be done easily with a mere offsetting. For example, अ is U+0905 while ਅ is U+0A05; क is U+0915 while ਕ is U+0A15.

However, the encoding doesn't represent the language in its true essence. Hindi, Sanskrit and most other Indian languages are centred around phonetic values. Hence the encoded token should ideally represent the entire sound rather than it being split into different symbols for vyanjana and maatra. We cannot figure out anything about the letter from its corresponding encoding. Which section of vyanjana it belongs to, whether it has a sweet or a pungent sound etc. The vyanjana symbols have a pre-added 'अ' (क् + अ = क). It is this conjoined sound which gets representation in Unicode rather than the plain क्. Our tool fixes this issue by creating a new representation that encapsulates the vyanjana and the swar in a single encoding.

3 Existing Work

A number of tools exist as of today for Sanskrit transliteration to other scripts and encodings. We present a brief survey of the same. Aksharamukha by Vinod Rajan, Sanscript by learnsanskrit.com and Online ITRANS are some of the tools currently used for transliteration in Sanskrit. Google Input is another tool that is used to transliterating Devanagari to English. Though Aksharamukha and Online ITRANS support the romanised forms of Sanskrit, none of the aforementioned tools manage to handle bilingual scenarios. Most of these (except Sanscript) also not open source and hence cannot be utilized by Sanskrit Developers.

International Phonetic Alphabet (IPA) is a renowned phonetic scheme. However, it has a number of representational and backward transliteration issues cause of being completely sound based. The nuktas don't share any correspondence to their roots gamma represents ga. The swar ऋ and रि have the same representation in IPA, making it impossible to differentiate them while translating back. Anuswar has multiple representations based on context, but none is unique to it (m, n, chandra). Visarga has the same representation as ha.

Due to these inefficacies of existing tools and phonetic scheme, we decided to create our own unified encoding schema. It also has a number of other desirable properties as described subsequently in the paper.

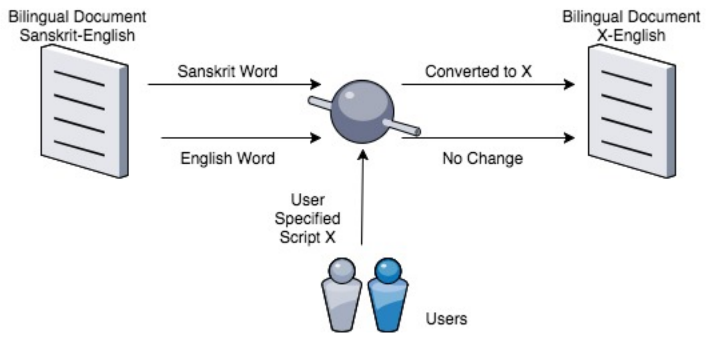


Figure 2: Model for Web-Based Applications

4 Design of the Transliterator

4.1 Use Cases

4.1.1 Web-based Applications

One of the foremost uses of our transliteration tool is it's utility for web-based applications. A number of websites nowadays serve the historical epics that were written in Sanskrit like the Gita and the Ramayana. Along with this, a lot of websites also provide an avenue for people to learn Sanskrit grammar, understand conjugation and splitting of words, along with explaining the various forms of Sanskrit verb roots. Such websites are as of now available only in the Devanagari script. Our tool can be used to transliterate these pages to a user defined script/encoding at the backend itself. Our model for this use case is defined in Figure 1. We insert our tool as a middle-ware between the backend and the frontend. The user specifies his required script/encoding on the frontend and all outgoing pages from the server pass through our tool while getting converted to that required script. The frontend then renders the converted HTML to the user for a seamless experience.

4.1.2 Bilingual Texts

Numerous Sanskrit texts have been modified to bilingual and trilingual texts through their translation to popular modern languages like English and Hindi. These works exist in a commentary form and incorporate massive amounts of code-switching. To represent any such text in a script different to that of its origin turns out to be an ordeal because the tool needs to conditionally perform the transliteration at a micro-level. This problem gets exacerbated when the Sanskrit verses are written using their Romanised form while the translation language is English. Figure 2 explains our model for this use case.

4.1.3 User Driven

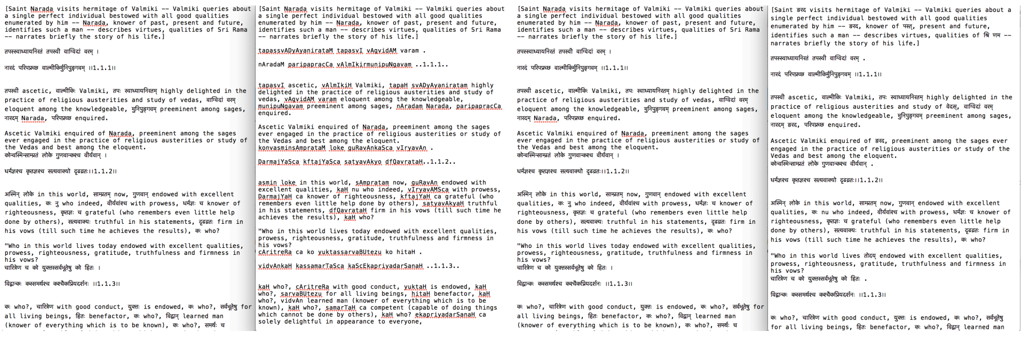
The third use for our tool is on the lines of Google Input tools. Our tool can allow a user to enter a line of Sanskrit (in any script) intertwined with English and will output the resulting sentence to the user after transliteration. This not only provides an unmatched amount of flexibility to the user, but also has abundant relevance in the growing age of multi-lingual social media.

4.2 Pipeline

Fragmentation: Splitting the given text into smaller fragments (words, sentences, paragraphs etc). The assumption shall be that the script and encoding remain same through these fragments if not through the entire text.

Script Detection: Figuring out the scripts and encodings for the various fragments through a Naive Bayes model.

Tokenisation: Splitting the fragment further into tokens, each of which represent a single sound. Similar to the concept of English syllables. ki will be seen as one single token under this model.



[Saint Narada visits hermitage of Valmiki -- Valmiki queries about a single perfect individual bestowed with all good qualities enumerated by him -- Narada, knower of past, present and future, identifies such a man -- describes virtues, qualities of Sri Rama -- narrates briefly the story of his life.]

तपस्स्वाध्यायनिरतं तपस्वी वाग्विदां वरम् ।

नारदं परिपप्रच्छ वाल्मीकिर्मुनिपुङ्गवम् ।।1.1.1।।

तपस्वी ascetic, वाल्मीकिः Valmiki, तपः स्वाध्यायनिरतम् highly delighted in the practice of religious austerities and study of vedas, वाग्विदां वरम् eloquent among the knowledgeable, मुनिपुङ्गवम् preeminent among sages, नारदम् Narada, परिपप्रच्छ enquired.

Ascetic Valmiki enquired of Narada, preeminent among the sages ever engaged in the practice of religious austerities or study of the Vedas and best among the eloquent.
कोन्वस्मिन्साम्प्रतं लोके गुणवान्कश्च वीर्यवान् ।

धर्मज्ञश्च कृतज्ञश्च सत्यवाक्यो दृढव्रतः।।1.1.2।।

अस्मिन् लोके in this world, साम्प्रतम् now, गुणवान् endowed with excellent qualities, कः नु who indeed, वीर्यवांश्च with prowess, धर्मज्ञः च knower of righteousness, कृतज्ञः च grateful (who remembers even little help done by others), सत्यवाक्यः truthful in his statements, दृढव्रतः firm in his vows (till such time he achieves the results), कः who?

"Who in this world lives today endowed with excellent qualities, prowess, righteousness, gratitude, truthfulness and firmness in his vows?
चारित्रेण च को युक्तस्सर्वभूतेषु को हितः ।

विद्वान्कः कस्समर्थश्च कश्चैकप्रियदर्शनः ।।1.1.3।।

कः who?, चारित्रेण with good conduct, युक्तः is endowed, कः who?, सर्वभूतेषु for all living beings, हितः benefactor, कः who?, विद्वान् learned man (knower of everything which is to be known), कः who?, समर्थः च

[Saint Narada visits hermitage of Valmiki -- Valmiki queries about a single perfect individual bestowed with all good qualities enumerated by him -- Narada, knower of past, present and future, identifies such a man -- describes virtues, qualities of Sri Rama -- narrates briefly the story of his life.]

tapassvADyAyanirataM tapasvI vAgvidAM varam .

nAradaM paripapracCa vAlmIkirmunipuNgavam ..1.1.1..

tapasvI ascetic, vAlmIkiH Valmiki, tapaH svADyAyaniratam highly delighted in the practice of religious austerities and study of vedas, vAgvidAM varam eloquent among the knowledgeable, munipuNgavam preeminent among sages, nAradam Narada, paripapracCa enquired.

Ascetic Valmiki enquired of Narada, preeminent among the sages ever engaged in the practice of religious austerities or study of the Vedas and best among the eloquent.
konvasminsAmprataM loke guRavAnkaSca vIryavAn .

DarmajYaSca kftajYaSca satyavAkyo dfQavrataH..1.1.2..

asmin loke in this world, sAmpratam now, guRavAn endowed with excellent qualities, kaH nu who indeed, vIryavAMSca with prowess, DarmajYaH ca knower of righteousness, kftajYaH ca grateful (who remembers even little help done by others), satyavAkyaH truthful in his statements, dfQavrataH firm in his vows (till such time he achieves the results), kaH who?

"Who in this world lives today endowed with excellent qualities, prowess, righteousness, gratitude, truthfulness and firmness in his vows?
cAritreRa ca ko yuktassarvaBUtezu ko hitaH .

vidvAnkaH kassamarTaSca kaScEkapriyadarSanaH ..1.1.3..

kaH who?, cAritreRa with good conduct, yuktaH is endowed, kaH who?, sarvaBUtezu for all living beings, hitaH benefactor, kaH who?, vidvAn learned man (knower of everything which is to be known), kaH who?, samarTaH ca competent (capable of doing things which cannot be done by others), kaH who? ekapriyadarSanaH ca solely delightful in appearance to everyone,

[Saint Narada visits hermitage of Valmiki -- Valmiki queries about a single perfect individual bestowed with all good qualities enumerated by him -- Narada, knower of past, present and future, identifies such a man -- describes virtues, qualities of Sri Rama -- narrates briefly the story of his life.]

तपस्स्वाध्यायनिरतं तपस्वी वाग्विदां वरम् ।

नारदं परिपप्रच्छ वाल्मीकिर्मुनिपुङ्गवम् ।।1.1.1।।

तपस्वी ascetic, वाल्मीकिः Valmiki, तपः स्वाध्यायनिरतम् highly delighted in the practice of religious austerities and study of vedas, वाग्विदां वरम् eloquent among the knowledgeable, मुनिपुङ्गवम् preeminent among sages, नारदम् Narada, परिपप्रच्छ enquired.

Ascetic Valmiki enquired of Narada, preeminent among the sages ever engaged in the practice of religious austerities or study of the Vedas and best among the eloquent.
कोन्वस्मिन्साम्प्रतं लोके गुणवान्कश्च वीर्यवान् ।

धर्मज्ञश्च कृतज्ञश्च सत्यवाक्यो दृढव्रतः।।1.1.2।।

अस्मिन् लोके in this world, साम्प्रतम् now, गुणवान् endowed with excellent qualities, कः नु who indeed, वीर्यवांश्च with prowess, धर्मज्ञः च knower of righteousness, कृतज्ञः च grateful (who remembers even little help done by others), सत्यवाक्यः truthful in his statements, दृढव्रतः firm in his vows (till such time he achieves the results), कः who?

"Who in this world lives today endowed with excellent qualities, prowess, righteousness, gratitude, truthfulness and firmness in his vows?
चारित्रेण च को युक्तस्सर्वभूतेषु को हितः ।

विद्वान्कः कस्समर्थश्च कश्चैकप्रियदर्शनः ।।1.1.3।।

कः who?, चारित्रेण with good conduct, युक्तः is endowed, कः who?, सर्वभूतेषु for all living beings, हितः benefactor, कः who?, विद्वान् learned man (knower of everything which is to be known), कः who?, समर्थः च

[Saint ङरद visits hermitage of Valmiki -- Valmiki queries about a single perfect individual bestowed with all good qualities enumerated by him -- ङरद, knower of पस्त्, present and future, identifies such a man -- describes virtues, qualities of श्रि णम -- narrates briefly the story of his life.]

तपस्स्वाध्यायनिरतं तपस्वी वाग्विदां वरम् .

नारदं परिपप्रच्छ वाल्मीकिर्मुनिपुङ्गवम् ।।1.1.1।।

तपस्वी ascetic, वाल्मीकिः Valmiki, तपः स्वाध्यायनिरतम् highly delighted in the practice of religious austerities and study of वेदस्, वाग्विदां वरम् eloquent among the knowledgeable, मुनिपुङ्गवम् preeminent among sages, नारदम् ङरद, परिपप्रच्छ enquired.

Ascetic Valmiki enquired of ङरद, preeminent among the sages ever engaged in the practice of religious austerities or study of the Vedas and best among the eloquent.
कोन्वस्मिन्साम्प्रतं लोके गुणवान्कश्च वीर्यवान् .

धर्मज्ञश्च कृतज्ञश्च सत्यवाक्यो दृढव्रतः।।1.1.2।।

अस्मिन् लोके in this world, साम्प्रतम् now, गुणवान् endowed with excellent qualities, कः nu who indeed, वीर्यवांश्च with prowess, धर्मज्ञः च knower of righteousness, कृतज्ञः च grateful (who remembers even little help done by others), सत्यवाक्यः truthful in his statements, दृढव्रतः firm in his vows (till such time he achieves the results), कः who?

"Who in this world lives तोदय् endowed with excellent qualities, prowess, righteousness, gratitude, truthfulness and firmness in his vows?
चारित्रेण च को युक्तस्सर्वभूतेषु को हितः .

विद्वान्कः कस्समर्थश्च कश्चैकप्रियदर्शनः ।।1.1.3।।

कः who?, चारित्रेण with good conduct, युक्तः is endowed, कः who?, सर्वभूतेषु for all living beings, हितः benefactor, कः who?, विद्वान् learned man

Figure 3: Transliteration of Bilingual Texts

Universalisation: Conversion of the token to the universal 16-bit encoding designed by us. Done through pre-populated hash maps.

Specification: Conversion of the universal encoding to the specified script using pre-populated hash maps.

4.3 Internal Representation

The encoding that we create is 16-bit unlike Unicode which requires 24-bits to represent most Indian characters. Initial 5 bits are for specifying the script (hence we can support 32). Next 6 bits are for the vyanjana (consonant). Last 5 bits are for the swar/maatra (vowel). In most cases, the vyanjana as well as the mantra both require 24-bits each in Unicode. We fit both in 16 bits. Each 16-bit code represents a specific sound, which can then be reverse mapped to a specified destination script.

With 33 consonants and 14 vowels, we can encode their permutations using just 9-bits versus the 11-bits that we currently are using. But, we preferred to use some extra bits so as to keep our representation clean and allow for the bits within themselves to represent certain nuances of the Sanskrit language.Our encoding respects the structure of the language as described by Panini and we can figure out important characteristics about the letter merely by looking at the encoding.

For the 5 bits of the Swaras, the second-last bit represents whether the vowel is a simple vowel (अ, इ, उ, ऋ, ऌ) or a dipthong/compound vowel (ए, ऐ, ओ, औ). The last bit of the Swaras represent the length of the vowel, long/dirgha vowels (आ, ई, ऊ, ॠ, ॡ, ए, ऐ, ओ, औ) will have their last bit as 1, while short/hrasva vowels (अ, इ, उ, ऋ, ऌ) will have their last bit as 0.

In the case of Vyanjana, the first 3 bits represent the source of origin of the letter. 000 refers to the throat as the source and the letters are called Gutturals (क्, ख्, ग्, घ्, ङ्, ह्). 001 refers to the palate and letters are called Palatals (च्, छ्, ज्, झ्, ञ्, य्, श्). 010 refers to the murdha and are called Retroflex letters (ट्, ठ्, ड्, ढ्, ण्, र्, ष्). 011 contains letters with source of origin as the teeth and are called Dentals (त्, थ्, द्, ध्, न्, ल्, स्). Lastly, 100 refers to the lips and the letters are called Labial (प्, फ्, ब्, भ्, म्, व्). 101, 110 and 111 are reserved for special symbols and accents.

As for the last 3 bits of Vyanjana, the first of these is 0 for stop-consonants (sparsa consonants) which means non-nasal, non-semivowel and non-sibilant consonants. The second of these bits represents voicing (whether or not the vocal chords vibrate in pronunciation). It is 1 for voiced/ghosa consonants like (ग्, घ्) while 0 for unvoiced/aghosa consonants like (क्, ख्). The last of these bits represents aspiration (a puff of air at the end of the pronunciation). It is 1 for aspirated/mahaprana consonants (ख्, घ्) while 0 for unaspirated/alpaprana consonants (क्, ग्).

5 Design of the Encoding Identifier

Differentiating English from Indian scripts, and those amongst themselves is easy as each uses a different alphabet with a different Unicode range. Hence, one can easily achieve a Word-level classifier with 100% accuracy.However, differentiating English from Romanized Sanskrit/Hindi requires learning. Specially to be able to do such classification at word-level.

Training Data: 1000 random Wiki pages for both English and Sanskrit. The Sanskrit ones were converted to SLP-1 using our universal encoder. We then parse out the irrelevant HTML meta-data and tags, stripping it down to just plain content.

Test Data: 100 more such random pages for both languages.

While learning, two dictionaries are maintained. The first dictionary compiles all seen complete words, while the other forms an occurence database of all possible substrings of length <= 10. The intuition is that certain sounds (or chain of sounds) occur more frequently in some languages then the others.

For a word, we define the Absolute Frequency of a word as the actual number of occurrences for that word for a given language in the training dataset. On the other hand, the Relative Frequency of a given word is defined as its fraction of occurrences in the given language versus all other languages under consideration. While classifying, if the word is a seen word and the Absolute as well as Relative frequency is above a pre-set threshold for a particular language, classify it as that. We use the Relative Frequency metric to account for mixed language nature of Wikipedia pages used as our dataset.

If we encounter an unseen word, we break the word into all possible substrings of length >= 2 and length <= 10. Subsequently, we find product(p(substr | lang)) over all substrings of word using the trained substring dictionary. This is our simplified version of the Naive Bayes model for the problem at hand. We classify a word to the language for which this metric turns out to be the maximum.

6 Results and Future Work

We tested our detection model on 100 random Sanskrit Wikipedia pages (after converting them to the 6 most popular romanisation schemes of SLP1, Velthuis, ITRANS, Harvard-Kyoto, ISO15919 and IAST). The resulting confusion matrices are shown in Table 1. As one can notice in the tables, we in general attain a high precision for English and a high recall for the romanised words.

Each scheme in Table 1 also has a corresponding baseline to compare our results with. For SLP1, this baseline was the existence of a capital letter in the middle of a word. For Velthuis, it was the existence of a full stop in the middle of a word or the existence of doubly repeated vowels. For ITRANS, the baseline was similar to Velthuis, with repeated 'L' and 'R' instead of full stop. For Harvard-Kyoto, we selected the baseline as capital in the middle of the word alongside repeated 'L' and 'R'. Lastly, for ISO15919 and IAST, it was kept as the existence of a letter beyond the simple English letters and punctuation within a word.

During our testing, we discovered that multiple English pronouns like 'Ram' or 'Yudhisthira' were getting classified as SLP-1 leading to a lower recall for English. In our opinion, such a misclassification aligns with the intention of the tool as it classifies the origin based on the prevalent sounds in the word. For Indian pronouns appropriated to English these sounds still remain similar to those of their Hindi roots, and hence rather should be classified as that. In our final evaluation, we manually removed such

Basline 67.2%	Pred English	Pred SLP-1	Recall
Actual English	73178	721	99.0%
Actual SLP-1	205	25012	99.2%
Precision	99.7%	97.2%	99.1%

(a) English vs SLP-1

Basline 58.6%	Pred English	Pred Velthuis	Recall
Actual English	72649	1250	98.3%
Actual Velthuis	860	24357	96.6%
Precision	98.8%	95.1%	97.9%

(b) English vs Velthuis

Basline 51.1%	Pred English	Pred ITRANS	Recall
Actual English	72778	1121	98.5%
Actual ITRANS	645	24572	97.4%
Precision	99.1%	95.6%	98.2%

(c) English vs ITRANS

Basline 68.5%	Pred English	Pred HK	Recall
Actual English	73269	630	99.1%
Actual HK	199	25018	99.2%
Precision	99.7%	97.5%	99.2%

(d) English vs Harvard-Kyoto

Basline 73.4%	Pred English	Pred ISO	Recall
Actual English	73576	323	99.6%
Actual ISO	94	25123	99.6%
Precision	99.9%	98.7%	99.6%

(e) English vs ISO15919

Basline 71.5%	Pred English	Pred IAST	Recall
Actual English	73368	531	99.3%
Actual IAST	111	25106	99.6%
Precision	99.8%	97.9%	99.4%

(f) English vs IAST

Table 1: Confusion Matrix of English vs Various Romanisation Schemata

ambiguous words both from the testing and well and training datasets.

We also tested our tool on a bilingual text test case by converting a extract from an English commentary on Ramayana from Gita-supersite to an mixture of SLP-1 and English. Subsequently, we converted the previous result back to Hindi and English to see its differences with the original text. As you can see in Figure 3, the transliteration from Devanagari-English to SLP1-English has a 100% accuracy due to our tool exploiting the difference in Unicode for the two scripts.

Our tool is available at `https://github.com/709nikhil/sanskrit-transliteration`. There is still some more future work that can be carried out on our tool. The primary one being heuristically breaking down word into syllables rather than substrings, to provide a stronger basis for the phoneme intuition. One could also go ahead and use a machine learning approach other than Naive Bayes, for example deep learning methods or CRFs. We could also incorporate contextual history into the transliteration to deal with the problem of incorrect classification of proper nouns.

References

Niraj Aswani and Robert J Gaizauskas. 2010. English-hindi transliteration using multiple similarity metrics. In LREC.

Peter Auer. 2013. Code-switching in conversation: Language, interaction and identity. Routledge.

Utsab Barman, Joachim Wagner, Grzegorz Chrupa□a, and Jennifer Foster. 2014. Dcu-uvt: Word-level language classification with code-mixed data. In Proceedings of the First Workshop on Computational Approaches to Code Switching, pages 127--132.

Shane Bergsma, Paul McNamee, Mossaab Bagdouri, Clayton Fink, and Theresa Wilson. 2012. Language identification for creating language-specific twitter collections. In Proceedings of the second workshop on language in social media, pages 65--74. Association for Computational Linguistics.

Akshar Bharati and Amba Kulkarni. 2007. Sanskrit and computational linguistics. In First International Sanskrit Computational Symposium, Hyderabad, pages 1--12.

Simon Carter, Wouter Weerkamp, and Manos Tsagkias. 2013. Microblog language identification: Overcoming the limitations of short, unedited and idiomatic text. Language Resources and Evaluation, 47(1):195--215.

William B Cavnar, John M Trenkle, et al. 1994. N-gram-based text categorization. Ann Arbor MI, 48113(2):161--175.

Gokul Chittaranjan, Yogarshi Vyas, Kalika Bali, and Monojit Choudhury. 2014. Word-level language identification using crf: Code-switching shared task report of msr india system. In Proceedings of The First Workshop on Computational Approaches to Code Switching, pages 73--79.

Sucheta Dasgupta. 2014. Home and away, sanskrit in resurgence mode. The Times of India.

Pawan Goyal and Gérard Huet. 2013. Completeness analysis of a sanskrit reader. In Proceedings, 5th International Symposium on Sanskrit Computational Linguistics. DK Printworld (P) Ltd, pages 130--171.

Pawan Goyal, Gérard P Huet, Amba P Kulkarni, Peter M Scharf, and Ralph Bunker. 2012. A distributed platform for sanskrit processing. In COLING, pages 1011--1028.

Harald Hammarström. 2007. A fine-grained model for language identification. In Proceedings of iNEWS-07 Workshop at SIGIR 2007, pages 14--20.

Gérard Huet, Amba Kulkarni, and Peter Scharf. 2009. Sanskrit computational linguistics. Lecture Notes in Computer Science, 5402.

Girish Nath Jha, Muktanand Agrawal, Sudhir K Mishra, Diwakar Mani, Diwakar Mishra, Manji Bhadra, Surjit K Singh, et al. 2009. Inflectional morphology analyzer for sanskrit. In Sanskrit computational linguistics, pages 219--238. Springer.

Dieter B Kapp and Thomas Malten. 1997. Report on the cologne sanskrit dictionary project. In 10th International Sanskrit Conference, Bangalore.

Ben King and Steven Abney. 2013. Labeling the languages of words in mixed-language documents using weakly supervised methods. In Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1110--1119.

Krishna Maheshwari. 2011. Features of sanskrit. Hindupedia.

Dong Nguyen and A Seza Do□ruöz. 2013. Word level language identification in online multilingual communication. In Proceedings of the 2013 conference on empirical methods in natural language processing, pages 857--862.

Peeta Basa Pati and AG Ramakrishnan. 2008. Word level multi-script identification. Pattern Recognition Letters, 29(9):1218--1229.

Sheldon Pollock. 2006. The language of the gods in the world of men: Sanskrit, culture, and power in premodern India. Univ of California Press.

Sanscript. `http://www.learnsanskrit.org/tools/sanscript`.

Srisa Chandra Vasu. 1897. The Ashtadhyayi of Panini.

Devanagari	Unicode	Velthius	SLP-1	ITRANS	Harvard-Kyoto	IAST	ISO-15919
अ	U+0905	a	a	a	a	a	a
आ	U+0906	aa	A	A/aa	A	ā	ā
इ	U+0907	i	i	i	i	i	i
ई	U+0908	ii	I	I/ii	I	ī	ī
उ	U+0909	u	u	u	u	u	u
ऊ	U+090A	uu	U	U/uu	U	ū	ū
ए	U+090F	e	e	e	e	e	ē
ऐ	U+0910	ai	E	ai	ai	ai	ai
ओ	U+0913	o	o	o	o	o	ō
औ	U+0914	au	O	au	au	au	au
ऋ	U+090B	.r	f	RRi/Rî	R	ṛ	r
ॠ	U+0960	.rr	F	RRI/RÎ	RR	ṝ	r̄
ऌ	U+090C	.l	x	LLi/Lî	lR	ḷ	l
ॡ	U+0961	.ll	X	LLI/LÎ	lRR	ḹ	l̄
अं	U+0902	.m	M	M/.n/.m	M	ṃ	ṁ
अः	U+0903	.h	H	H	H	ḥ	ḥ
अँ	U+0904		~	.N			m
ऽ	U+093D	.a	’	.a	’	’	’
क	U+0915	ka	ka	ka	ka	ka	ka
ख	U+0916	kha	Ka	kha	kha	kha	kha
ग	U+0917	ga	ga	ga	ga	ga	ga
घ	U+0918	gha	Ga	gha	gha	gha	gha
ङ	U+0919	”na	Na	Na	Ga	ṅa	ṅa
च	U+091A	ca	ca	cha	ca	ca	ca
छ	U+091B	cha	Ca	Cha	cha	cha	cha
ज	U+091C	ja	ja	ja	ja	ja	ja
झ	U+091D	jha	Ja	jha	jha	jha	jha
ञ	U+091E	na	Ya	na	Ja	ña	ña
ट	U+091F	.ta	wa	Ta	Ta	ṭa	ṭa
ठ	U+0920	.tha	Wa	Tha	Tha	ṭha	ṭha
ड	U+0921	.da	qa	Da	Da	ḍa	ḍa
ढ	U+0922	.dha	Qa	Dha	Dha	ḍha	ḍha
ण	U+0923	.na	Ra	Na	Na	ṇa	ṇa
त	U+0924	ta	ta	ta	ta	ta	Ta
थ	U+0925	tha	Ta	tha	tha	tha	Tha
द	U+0926	da	da	da	da	da	Da
ध	U+0927	dha	Da	dha	dha	dha	Dha
न	U+0928	na	na	na	na	na	na
प	U+092A	pa	pa	pa	pa	pa	pa
फ	U+092B	pha	Pa	pha	pha	pha	pha
ब	U+092C	ba	ba	ba	ba	ba	ba
भ	U+092D	bha	Ba	bha	bha	bha	bha
म	U+092E	ma	ma	ma	ma	ma	ma
य	U+092F	ya	ya	ya	ya	ya	ya
र	U+0930	ra	ra	ra	ra	ra	ra
ल	U+0932	la	la	la	la	la	la
व	U+0935	va	va	va/wa	va	va	va
श	U+0936	”sa	Sa	sha	za	śa	śa
ष	U+0937	.sa	za	Sha	Sa	ṣa	ṣa
स	U+0938	sa	sa	sa	sa	sa	sa
ह	U+0939	ha	ha	ha	ha	ha	Ha

Table 2: Comparison of various Devanagari Romanisations