NELIS - Named Entity and Language Identification System: Shared Task System Description

Rampreeth Ethiraj[1], Sampath Shanmugam[2], Gowri Srinivasa[3]
PES Center for Pattern Recognition
PESIT Bangalore South Campus
Bengaluru, Karnataka
India
[1]ethirajrampreeth@gmail.com, [2]sampath_shanmugam@outlook.com,
[3]gsrinivasa@pes.edu

Navneet Sinha
Rochester Institute of Technology
Rochester, New York
USA
navneet.sinha27@gmail.com

ABSTRACT
This paper proposes a simple and elegant solution for language identification and named entity (NE) recognition at a word level, as a part of Subtask-1: Query Word Labeling of FIRE 2015. Given any query q_1:w_1 w_2 w_3 … w_n in Roman script, the task calls for labeling words of the query as English (En) or a member of *L*, where *L* = {Bengali (Bn), Gujarati (Gu), Hindi (Hi), Kannada (Kn), Malayalam (Ml), Marathi (Mr), Tamil (Ta), Telugu (Te)}. The approach presented in this paper uses the combination of a dictionary lookup with a Naïve Bayes classifier trained over character n-grams. Also, we devise an algorithm to resolve ambiguities between languages, for any given word in a query. Our system achieved impressive f-measure scores of 85-90% in four languages and 74-80% in another four languages.

Keywords
Language Identification, N-grams, Naïve Bayes classifier

1. INTRODUCTION
India's heritage in languages is one of the richest in the world and is also known as the "Museum of Languages". India is a multi-language, multi-script country, with 22 official languages. A large number of these languages are written using indigenous scripts. However, often websites and user generated content such as tweets and blogs in these languages are written using Roman script [1] due to various social, cultural and technological reasons. This paper presents an approach to analyze a sentence written in En and a transliterated language *L*, where *L* = {Bn, Gu, Hi, Kn, Ml, Mr, Ta, Te}, adopting the Roman script, from sources such as tweets, blogs and user-generated messages and button down the language every word belongs to.

The philosophy of this approach was inspired partially by how humans identify languages of words. First, if the word is a part of their vocabulary, then they know the language of the word. If the word is unfamiliar to them, then they tend to make a guess, based on the structure of the word. Finally, if they are given a sentence and have managed to decode the language of a few words, then they can make a fairly accurate guess about the language of the unknown words as well. A close analogy can be drawn between the above and the approach suggested in this paper; the human language vocabulary is equivalent to the language dictionaries and the guess made based on the features of the word is performed by the Naïve Bayes classifier, using n-gram as features. A logical method for disambiguation is suggested in this paper.

2. DATASETS
The core of the system was building strong dictionaries for each language. The wordlists used to compile the dictionaries are listed in Table 1.

Table 1. Primary sources used to prepare dictionaries.

Class	Source
Bn, Hi, Gu	FIRE 2013 Dataset [2]
En	Mieliestronk's word list http://www.mieliestronk.com/wordlist.html + FIRE 2013 Dataset
MIX	FIRE 2015 Dataset
NE	FIRE 2015 Dataset
Bn, Gu, Kn, Ml, Ta, Te	List of most frequently used English words [3] were translated and transliterated.

The most frequently used words in En were translated into their respective Indian language equivalents, using Google's online translation service[1]. But the translated words were all in their native scripts. These had to be transliterated into their Roman equivalents. The process of phonetically representing the words of a language in a non-native script is called transliteration [4]. Baraha Software[2] was used to transliterate these words into their Roman script equivalents.

While this sufficed for En and Hi, the data collected was not enough for accurate classification of other languages. Thus, in addition to these word lists, mining of data from other sources was necessary to account for various spelling variations [5] and also to capture the commonly used words of each language. These secondary sources include song lyrics, common SMS messages, and 'learn to speak' websites found online. Even shorthand notations of various words were effectively captured from these sources.

For example, consider Gu. '*che*' is also sometimes spelt as '*6e*' .

We manually extracted language words in Roman form from these secondary sources, cleaned them and keyed them into the dictionaries. Table 2 lists these secondary sources.

Comprehensive dictionaries were hence manually formed for each language. Table 3 lists the final sizes of all language dictionaries.

[1] https://translate.google.com/

[2] http://www.baraha.com

Organization of dictionaries:

Each language dictionary was divided into sub-dictionaries based on the starting character, sorted alphabetically, to speed up the process of dictionary lookup. For example, all tokens of a language that started with 'a' would be grouped together.

Table 2. Secondary sources used to prepare dictionaries.

Class	Source
Bn, Gu, Kn, Ml, Mr, Ta, Te	Song Lyrics Kn, Mr, Te: http://www.hindilyrics.net/ Gu: http://songslyricsever.blogspot.com/p/blog-page_9289.html Ml: http://www.malayalamsonglyrics.net Bn: http://www.lyricsbangla.com Ta: http://www.paadalvarigal.com/
Bn, Gu, Kn, Ml, Mr, Ta, Te	SMS messages and 'learn to speak' websites. http://www.funbull.com/sms/sms-jokes.asp http://www.omniglot.com/language/phrases/langs.html
X	Commonly used SMS abbreviations. http://www.connexin.net/internet-acronyms.html
NE	Common names of people, places, organizations and brands. https://bitbucket.org/happyalu/corpus_indian_names/downloads http://simhanaidu.blogspot.in/2013/01/text-list-of-indian-cities-alphabetical.html http://www.elections.in/political-parties-in-india/ http://business.mapsofindia.com/top-brands-india/

Table 3. Final sizes of all language dictionaries

Language	Dictionary Size (in words)
En	97271
Hi	26094
Ta	23992
Te	25472
Bn	19573
Mr	10564
Gu	20729
Ml	22219
Kn	32479

3. APPROACH

Problem Statement:

Suppose that a query is given in Roman script, the task is to label the words as En or a member of *L*. Assumptions to be made:

1. The words of a single query usually come from 1 or 2 languages and very rarely from 3 languages.
2. In case of mixed language queries, one of the languages is either En or Hi.

The approach is divided into two sections; Section 3.1 explains the process of classification of tokens, while Section 3.2 elaborates on the process of disambiguation. Figure 1 depicts the overall process.

3.1 Classification of Tokens

The system built to demonstrate this approach was written entirely in Python, using the NLTK package[3] for processing and classification. The test file provided consisted of utterances (sentences or queries). The system read the input file utterance by utterance, and each utterance was tagged token (word) by token, sequentially. Section 3.1.1 explains the tagging of X tokens with regular expressions, Section 3.1.2 explains process of tagging of language tokens. At the end of the process, an annotated output file was generated.

3.1.1 Regular Expression based Tagging

Regular Expressions were used to match X tokens [6]. Table 4 shows the expressions used and their class. The X dictionary was also referenced in case none of the expressions matched the token.

3.1.2 Language Tagging

To tag language tokens, the combination of dictionary lookup and Naïve Bayes classifier were used. The subsections below explain the process of tagging language tokens. The techniques were combined and used, sequentially.

3.1.2.1 Dictionary Lookup and Tagging

Dictionaries of all language were looked up each time, if the token has not been already tagged as X, MIX or NE. Three cases could arise:

Case 1:

The token belongs to exactly one language. Hence tag as this language.

Case 2:

[3] http://www.nltk.org

The token belongs to more than one language. Tag as ambiguous, along with the set of languages causing the ambiguity.

Case 3:

The token is not found in any of the language dictionaries. Use the Naïve Bayes classifier to guess the language, as explained in Section 3.1.2.2.

After all tokens had been tagged by the dictionary, an aggregation of the number of occurrences of each language tag was performed. This is used later while trying to resolve ambiguity.

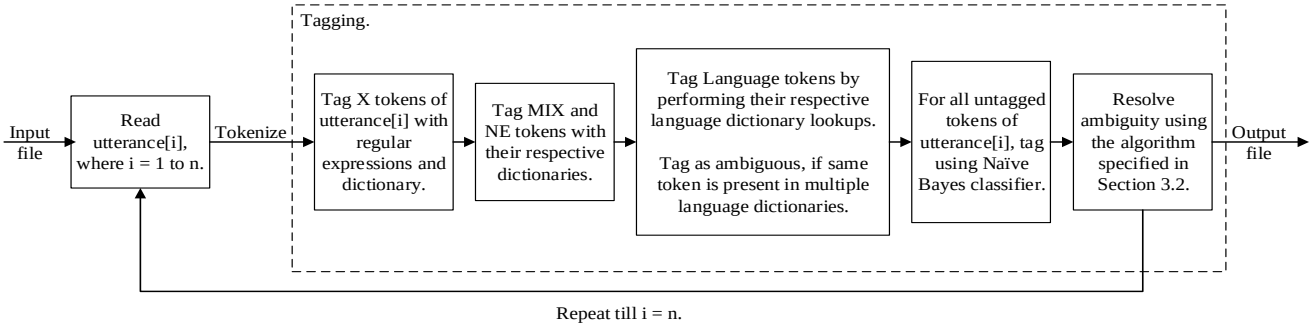

Figure 1. Overall process of tagging, from input to output.

Table 4. Regular Expressions used to tag X

Regular Expression	Class
r'[\.\=\:\;\,\#\@\(\)\`\~\$*\!\?\"\+\- \\\/|\{\}\[\]_\<\>\%\&]+'	X
r'[0-9]+'	X
r'[a-zA-Z]+[\@]+[a-zA-Z\.]*'	X
r'http+'	X
r'www.[A-Za-z0-9]+.com'	X
r'[A-Za-z0-9]+.com'	X
r'[0-9]+[tT][hH]'	X
r'[0-9]*[1]+[sS][tT]'	X
r'[0-9]*[2][nN][dD]'	X
r'[0-9]*[3][rR][dD]'	X
r'[^a-zA-z]' & length of token = 1	X

3.1.2.2 Naïve Bayes Classifier and Tagging

An inherently multiclass Naïve Bayes classifier, from the NLTK package was trained specifically for language identification. Each language $l \in L$ is a class. While training, the frequencies of co-occurrences of character n-grams in the language dictionaries prepared in Section 2 were analyzed. An n-gram is an n-character slice of a longer string [7]. A frequency distribution of character 2-gram, 3-gram, 4-gram and 5-gram was studied and used for the purpose of training the classifier.

$$lang = \arg\max_{l \in L} \left(\frac{(P(t \mid l)P(l))}{P(t)} \right)$$

where *lang* is the language of a given token, *t* is the token and *l* is a language in *L*.

Those tokens that were not tagged after the dictionary look up were tagged by the Naïve Bayes classifier. After all tokens had been tagged by the classifier, an aggregation of the number of occurrences of each language tag was performed. But this time, the number of occurrences of each language was multiplied by a certain specific weight. This weight was based on the accuracy of the classifier for that particular language. These were added with the previously computed values for each language while performing language dictionary lookups in Section 3.1.2.1

3.2 Further Processing and Disambiguation

Disambiguation of words belonging to multiple languages tends to be a challenge, unless the context of the utterance is known. In cases where utterances were bilingual, based on observation of the training set, we concluded that it is more probable for En to be a part of the bilingual utterance.

To begin the process, we perform yet another count, but this time exclusively for ambiguous tokens. A count of the number of occurrences for each language was computed and was multiplied by a weight. Let this weight be $size_l$ for any given language *l* in *L*, where

$$size_l = \left(\frac{\text{size of dictionary of that language } l}{\text{total size of all dictionaries in L-\{En\} put together}} \right)$$

En was not taken into account while computing the total sum, because of the large size of the dictionary. These newly computed scores were added to the scores computed previously in Section 3.1.2.2 for each language and were used determine the language(s) of the utterance. The language with the maximum score is ranked highest.

Hence the challenge was to be able to identify either a language or a pair of languages for each utterance. This was done by identifying the most frequently occurring Indian language, say *lang*, in an utterance, and the count of En in this utterance, as computed previously. The steps involved in resolving ambiguity in an utterance is as follows.

Step 1:

All those unambiguous tokens that belonged to neither *lang* nor En were converted to *lang*. This assumption was made given the strength of the En dictionary, as the probability of a new word belonging to En, given that it is not in the En dictionary, is low.

Step 2:

All ambiguous tokens, where the ambiguity was between En and another language or a set of languages, and *lang* is absent, were converted to En.

Step 3:

All ambiguous tokens, where the ambiguity was between *lang* and another language or a set of languages, were converted to *lang*.

Step 4:

For all ambiguous tokens that were not disambiguated in the previous steps, the following was followed:

If the token is not the first token in the utterance and the previous token is a language token, then the token will be of the same language as the previous token.

Else If the next token is a language token, then the current token will be of the same language.

Else, tag as En.

This scheme worked by identifying the overall language(s) of the utterance and then narrowing it down to the language of the individual token, for disambiguation.

4. RESULTS

A single run was submitted for the subtask and the results are summarized in the Table 5 and Table 6.

Table 5. Summary of the scores obtained for each class

Class	Strict Precision	Strict Recall	Strict f-measure
MIX	0	0	0
NE	0.645	0.326	0.433
X	0.952	0.941	0.947
Bn	0.795	0.921	0.853
En	0.898	0.852	0.874
Gu	0.270	0.490	0.349
Hi	0.713	0.841	0.771
Kn	0.937	0.814	0.871
Ml	0.675	0.830	0.744
Mr	0.808	0.774	0.791
Ta	0.912	0.872	0.891
Te	0.774	0.778	0.777

Table 6. Summary of the overall scores obtained

Measure	Run-1
TokensAccuracy	82.715
UtterancesAccuracy	26.389
Average F-measure	0.692
Weighted F-measure	0.829

5. ERROR ANALYSIS

This system yields very promising results for word level language identification and named entity recognition. Bn, En, Kn, Ta all have f-measures above 85%. Similarly, the remaining languages with the exception of Gu have f-measures above 74%.

Errors during translation and transliteration are to be accounted for. The accuracy of Gu was comparatively low. Upon detailed analysis, it was observed that various spelling variations could not be accounted for, neither in the dictionaries, nor while training. Also, much ambiguity existed between Hi and Gu. Because Hi words are more frequently occurring, the system is biased towards Hi in such ambiguous situations. This made it particularly very difficult to identify correctly Gu in utterances of short length.

For example, from the training set provided:

praan ni antim yatra

Here *praan*, *antim* and *yatra* are all Hi words too.

It fails to tag mix words in the test dataset due to the presence of MIX tokens in specific language dictionaries in the training data.

For example, *account-la*, where *account* is En, *la* is Ta, is in the Ta dictionary. This explains the low scores for MIX.

6. CONCLUSION AND FUTURE SCOPE

In this paper, we present the brief synopsis of a methodology to classify query words into their respective languages. The methodology involves a dictionary lookup networked with a Naïve Bayes classifier to accomplish the task. Usage of word level n-grams as a feature to the Naïve Bayes classifier can be experimented with. A new approach to identify and tag MIX tokens will have to be devised. Furthermore, the accuracy of Gu and the overall accuracy of the system can be upgraded by devising a new technique to handle the indeterminateness between Hi and Gu.

7. REFERENCES

[1] Umair Z. Ahmed, Kalika Bali, Monojit Choudury, Sowmya VB. Challenges in Designing Input Method Editors for Indian Languages: The Role of Word-Origin and Context. In *Proceedings of the WTIM*, pages 1-9, 2011.

[2] FIRE 2013 Dataset. Datasets for FIRE 2013. URL: http://cse.iitkgp.ac.in/resgrp/cnerg/qa/fire13translit/index.html. *Last accessed*: October 5, 2015.

[3] first20hours. google-10000-english/20k.txt. URL: https://github.com/first20hours/google-10000-english/blob/master/20k.txt. *Last accessed:* October 5, 2015.

[4] Kevin Knight, Jonathan Graehl. "Machine Transliteration". *Computational Linguistics*, pages 599-612, 1998.

[5] Royal Denzil Sequiera, Shashank S. Rao, Shambavi B R. Word - Level Language Identification and Back Transliteration of Romanized Text: A Shared Task Report by BMSCE. *Shared Task System Description in MSRI FIRE Working Notes*, 2014.

[6] Navneet Sinha, Gowri Srinivasa. Hindi-English Language Identification, Named Entity Recognition and Back Transliteration: Shared Task System Description. *Shared Task System Description in MSRI FIRE Working Notes*, 2014.

[7] William B. Cavnar, John M. Trenkle. N-Gram-Based Text Categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161-169, 1994.