Romanagari Detection in Twitter

14 Oct 2015

Hrishikesh Terdalkar - Shubhangi Agarwal

Motivation

- Why Twitter?
- Most NLP techniques deal with English text only
- Tweets are often of the form:

“Yeh kaisi field placings lagayi hain? Powerplay mein slip? Via @ARangarajan1972 #IndvsPak ”

Romanagari = Noise

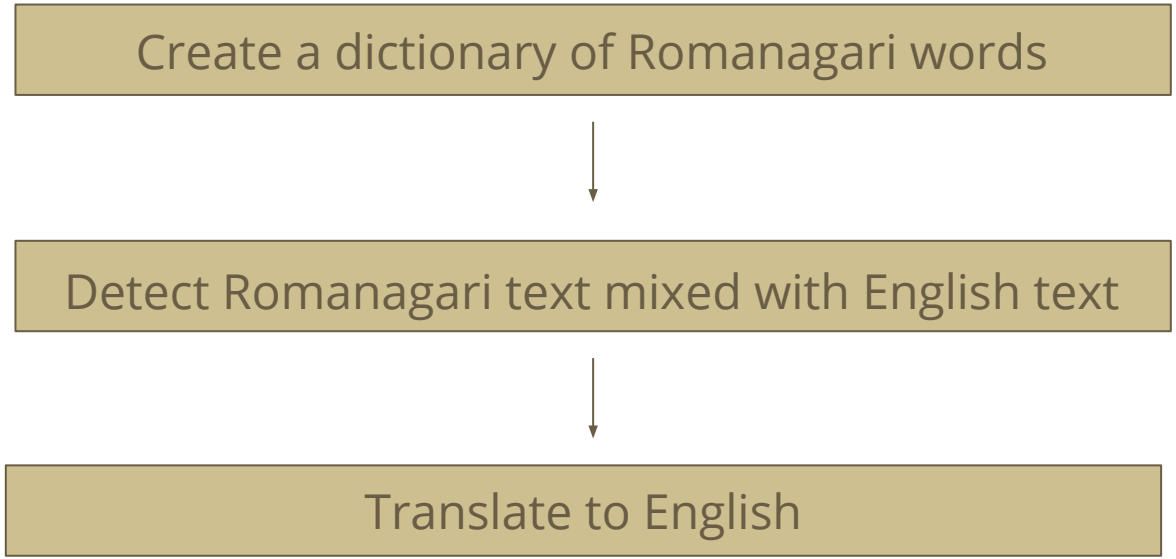
Goal

- Collect and create a quality tweet-dataset containing Romanagari words
- Romanagari Text Detection
- (possibly) Translate to English language

Languages Targeted

- Hindi

Steps


Create a dictionary of Romanagari words
Detect Romanagari text mixed with English text
Translate to English

Sounds easy?!

Challenges

1. Data Collection
    i. Search terms
    ii. Noise (different languages)
    iii. Disambiguation (polysemy in Hindi and English)

2. Detect and differentiate between English and Romanagari text
    i. Phonetic typing
    ii. SMS language
    iii. Spelling errors
    iv. Disambiguation

Challenges

3. Handle commonly occurring inflections in the social media text
    i. whatttttttt!, whennnn??, kyunnnn??
    ii. mann, bool, bol

4. And many more (yet to be encountered)

Approach

1. Data
    i. Collection
        - ➢ Frequent Romanagari words
        - ➢ Tweepy
        - ➢ SMS language
    ii. Synthetic Generation

2. Language detection/correction
    i. Tools available (PyEnchant, langid, langdetect, guess-language etc)

3. Almost phonetic representations
    i. Metaphone
    ii. Double Metaphone
    iii. Soundex

    - ➢ Also used for Romanagari text detection

Strategies

- Find frequently used Romanagari words in tweets/social media. (Different from “most frequent” Hindi words from other corpora such as books / wiki)

- Try to obtain annotated-datasets from social media such as facebook from existing papers and frequency analysis on this smaller “spoken-hindi” dataset.

- Context analysis (if possible)
    - ➢ n-grams

So far..

- ➔ Python
- ➔ Twitter collection
  - most frequent hindi words as FILTER
  - low success rate on tweets + lot of noise
  - explored synthetic generation [3]
- ➔ Exploration of existing classifiers
  - *PyEnchant*: a spellchecking library for Python based on Enchant
  - *langdetect*: python implementation of “language-detection” Java library
  - *langid*: language identification, n-gram, 97 languages, scores for multiple languages
- ➔ Soundex / Metaphone Experiments

Soundex vs Double Metaphone

“kyun”

```
>>> s("kyun")
'K50000'
>>> s("kyunn")
'K50000'
>>> s("kyunnnnn")
'K50000'
```

```
>>> doublemetaphone("kyun")
('KN', '')
>>> doublemetaphone("kyunn")
('KN', '')
>>> doublemetaphone("kyunnnnn")
('KNNN', '')
```

“haan”

```
>>> s("haan")
'H50000'
>>> s("haaaannn")
'H50000'
```

```
>>> doublemetaphone("haan")
('HN', '')
>>> doublemetaphone("haaaannn")
('HNN', '')
```

“what”

```
>>> s("what")
'W30000'
>>> s("wwhaaattt")
'W30000'
```

```
>>> doublemetaphone("what")
('AT', '')
>>> doublemetaphone("wwhaaattt")
('TT', '')
```

Soundex vs Double Metaphone

"burp"

```
>>> s("burp")
'B61000'
>>> s("burrrrppp")
'B61000'
```

```
>>> doublemetaphone("burp")
('PRP', '')
>>> doublemetaphone("burrrrppp")
('PRRPP', '')
```

"boom"

```
>>> s("boooooooooom")
'B50000'
>>> s("boom")
'B50000'
>>> s("boon")
'B50000'
```

```
>>> doublemetaphone("boooooooooom")
('PM', '')
>>> doublemetaphone("boom")
('PM', '')
>>> doublemetaphone("boon")
('PN', '')
```

"lol"

```
>>> s("lol")
'L40000'
>>> s("lollll")
'L40000'
```

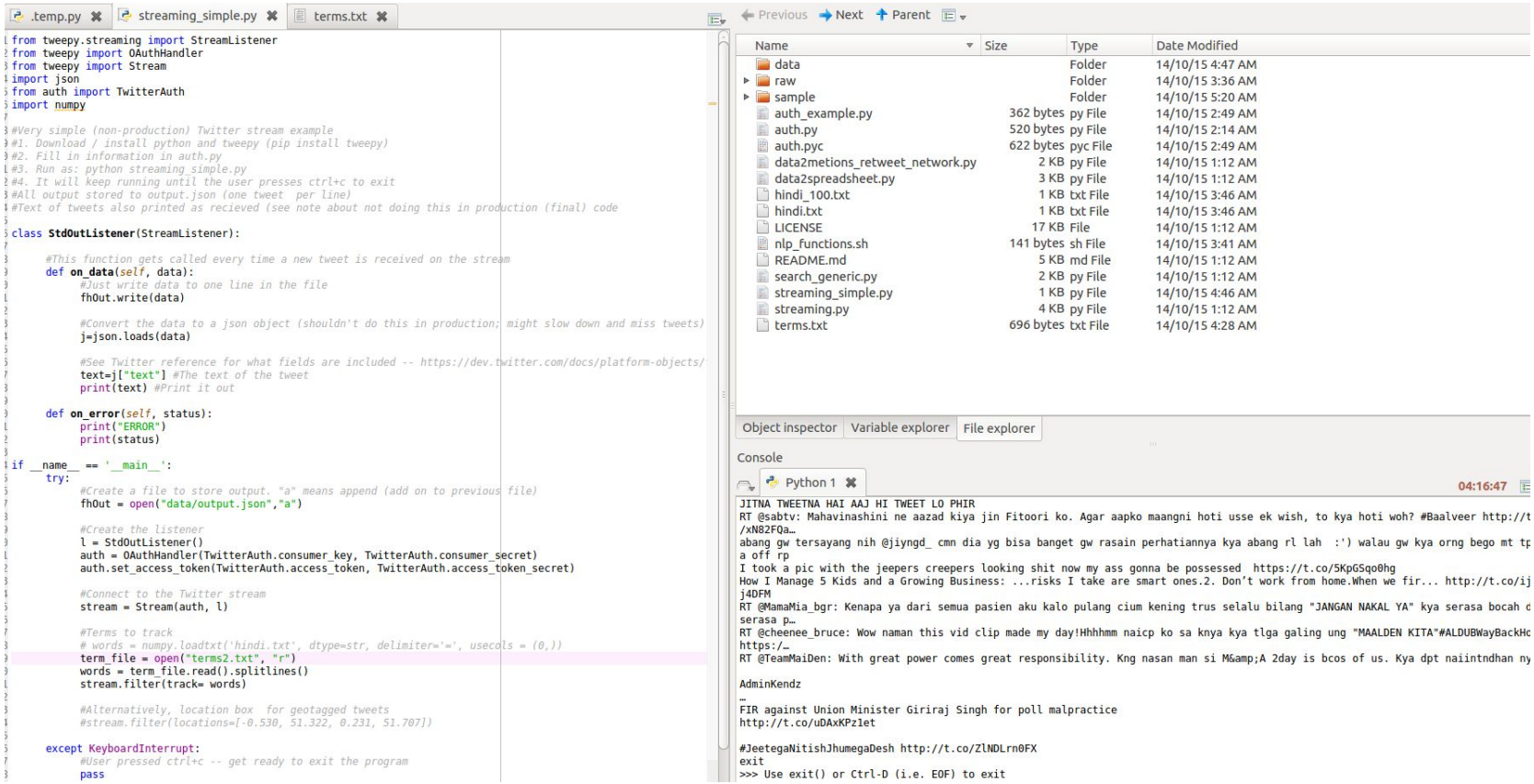
```
>>> doublemetaphone("lol")
('LL', '')
>>> doublemetaphone("lollll")
('LLL', '')
```

"ah / oh"

```
>>> s("ah")
'A00000'
>>> s("aaahhh")
'A00000'
>>> s("ohhh")
'O00000'
>>> s("ooohhh")
'O00000'
```

```
>>> doublemetaphone("ah")
('A', '')
>>> doublemetaphone("aaahhh")
('A', '')
>>> doublemetaphone("ohhh")
('A', '')
>>> doublemetaphone("ooohhh")
('A', '')
```

Tweet Collection



Plan

- ➔ Better dataset collection strategies
- ➔ Better synthetic generation than mentioned in [3]
- ➔ Perform experiments to test feasibility of Soundex/Metaphone for Hindi
- ➔ Pre-processing tweets followed by language identifiers with modification
- ➔ Compose a list of Hindi-specific disambiguation rules
- ➔ Detect Romanagari words
- ➔ Annotate / Attach English meaning to Romanagari words

References

1. Barman, Utsab, et al. "Code Mixing: A Challenge for Language Identification in the Language of Social Media." *EMNLP 2014* (2014): 13.
2. Gella, Spandana, Jatin Sharma, and Kalika Bali. "Query word labeling and back transliteration for indian languages: Shared task system description." *FIRE Working Notes* (2013).
3. Gella, Spandana, Kalika Bali, and Monojit Choudhury. ""ye word kis lang ka hai bhai?" Testing the Limits of Word level Language Identification."
4. Das, Amitava, and Björn Gambäck. "Identifying Languages at the Word Level in Code-Mixed Indian Social Media Text." *Proceedings of the 11th International Conference on Natural Language Processing, Goa, India*. 2014.
5. Han, Bo, and Timothy Baldwin. "Lexical normalisation of short text messages: Makn sens a# twitter." *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 2011.
6. Proceedings of Social india 2014
7. Tweepy: https://github.com/tweepy/tweepy
8. Chaware, Sandeep, and Srikantha Rao. "Rule-Based Phonetic Matching Approach for Hindi and Marathi." Computer Science & Engineering, 1.3 (2011), AIRCC

Questions?

Thank You