Columbia-Jadavpur submission for EMNLP 2016 Code-Switching Workshop Shared Task: System description

Arunavha Chanda
a.chanda@columbia.edu
Columbia University

Dipankar Das
dipankar.dipnil2005@gmail.com
Jadavpur University

Chandan Mazumdar
chandanm@cse.jdvu.ac.in
Jadavpur University

Abstract

We describe our present system for language identification as a part of the EMNLP 2016 Shared Task. We were provided with the Spanish-English corpus composed of tweets. We have employed a predictor-corrector algorithm to accomplish the goals of this shared task and analyzed the results obtained.

1 Introduction

Code-mixing, a phenomenon in linguistics, is exhibited by multi-lingual people. Any utterance in which the speaker makes use of the grammar and lexicon of more than one language is said to have undergone code-mixing or code-switching (Appel and Muysken, 2005).

English is considered the primary language of use, as well as the most widely used language on the internet, accounting for around 53.6% content language of websites (W3Techs, 2015). It may be a bit of surprise that the value isn't higher. However, the statistics on social media re-inforce this idea, since around half of the messages on Twitter are in non-English languages (Schroeder, 2010).

In contrast to English, multilingual people tend to communicate in several of the languages that they know. This is because of several reasons: some multilingual speakers feel higher level of comfort in their native language than in English; some conversational topics are more fluid in a particular language and some expressions convey the message properly only in one's native language.

In this paper, we describe our system based on predictor-corrector algorithm as part of the shared task of EMNLP 2016 Code-Switching Workshop. The system has been applied on the English-Spanish code-mixed corps of tweets. Several lexicons were employed along with some rules into the system, the results were obtained and discussed in detail.

Section 2 describes the conference task description, Section 3 deals with the tools and techniques we used, Section 4 describes the system functioning, Section 5 talks about the results obtained and finally, Section 6 closes with a discussion.

2 Task description

The EMNLP 2016 Code-Switching Workshop[1] included a Shared Task on two language pairs: (1) English-Spanish and (2) Modern Standard Arabic-Arabic dialects. In the present attempt, We worked only on the English-Spanish task. The task organizers provided a corpora composed from code-switched tweets on which annotation had to be done using the following labels:

1. **lang1**: Language 1 of the pair- English in our case. We use this if the word is undoubtedly used in English in the given context.

2. **lang2**: Language 2 of the pair- Spanish for us. It is same as lang1 and we use this as this word is undoubtedly used as Spanish.

3. **mixed**: Mixed words for the words that are composed of both the languages. An example given was "*Snapchateame*", in which "Snapchat" was used from English and "*eame*" was from Spanish.

[1] http://care4lang1.seas.gwu.edu/cs2/call.html

Proceedings of the Second Workshop on Computational Approaches to Code Switching, pages 112–115,
Austin, TX, November 1, 2016.

4. **NE**: Named Entities- used for proper nouns like people, places, organizations, locations, titles and such.

5. **ambiguous**: Ambiguous words that exist in both English and Spanish and suh words are hard to be clarified based on the context given.

6. **FW**: Foreign words, which do not appear either in English or in Spanish, but exist in another language and used in that context.

7. **UNK**: Unknown words which do not fit any of the above categories and is unrecognizable.

8. **Other**: Numbers, symbols, emojis, URLs and anything else that is not a "word". However, the words beginning with a "hashtag" (#) are treated as other tag.

The tweets were provided in terms of sentences and we were asked to develop a system that would annotate every token in the entire corpus of tweets as one of the given eight labels.

3 Tools and Techniques Used

3.1 Lexicons used

The dictionaries we used are the following:

1. **English dictionary**: We use the Python Enchant library[2] for checking the English words and their existence in the dictionary. We also create a slang dictionary of our own containing colloquial English text words such as "LOL" and "gr8". We collected the lexicons from the works of researchers at the University of Melbourne and University of Texas at Dallas (Han et al., 2012; Liu et al., 2011; Liu et al., 2012).

2. **Spanish dictionary**: We use the Python Enchant library once again for checking of the words' existence in the Spanish dictionary.

3. **Foreign dictionaries**: We also use the Italian, French, Portuguese (Brazil), Portuguese (Portugal) and German dictionaries from Python Enchant to check for words' existence. Since the geographical spread is given, any person with code-switching possibility between English and Spanish would most likely borrow words from one of these languages.

4. **Stanford Named Entity Recognizer**: For identifying the named entities, we used the Stanford NER (Named Entity Recognizer) (Finkel et al., 2005) and it's Python interface in the nltk library[3].

3.2 Algorithm

3.2.1 Word Slicing

For identifying the mixed words, we use a word-slicing algorithm. It consists of the following steps:

1. We keep slicing a word into two parts of varying lengths. For example, for "abcde", we would obtain four splits:
   - "a" and "bcde"
   - "ab" and "cde"
   - "abc" and "de"
   - "abcd" and "e"
2. For each of these splits, we check if one part is present in the English dictionary and one part appears in the Spanish dictionary. In such cases, it would be declared as a mixed word. For example, if "abc" was identified as an English word and "de" was a Spanish word, "abcde" would be declared a mixed word.

3.2.2 Predictor-Corrector algorithm

We use this algorithm for the words that are present in both the English and Spanish dictionaries.

- Prediction: During initial tagging, if a word is present in both the dictionaries, it is tagged as "both".
- Correction: During the second round, we return to the point of words that are tagged "both" and if both the words on either side (or the adjacent one if at the beginning or end) are in the same language, it is corrected to that language, otherwise marked as ambiguous.

This way, our predictor-corrector method helps us to achieve better accuracy for identifying the ambiguous words.

[2]https://pypi.python.org/pypi/pyenchant/

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

4 System Description

We take every tweet at a time and come down to the word level tagging before moving to the next tweet.

4.1 Dictionary words

- For every word, we first strip it of a "hashtag" (#), if there. Next, we run the Stanford Named Entity Recognizer[4]. The words identified as a Named Entity are tagged **"NE"** within a sentence.

- Before adopting our dictionary checking module, we check whether the token is all-word or it contains any punctuation mark/special character or number in it or not. If it does, we label it as **other**. Otherwise, we advance to the next step.

- Next, we check for the word's presence in the English and Spanish dictionaries. Based on their presence or absence in either dictionary, we take action:
  - If the word is present in the English dictionary and absent in the Spanish dictionary or vice-versa, it is immediately tagged **lang1** or **lang2** respectively.
  - If the word is present in both the English and Spanish dictionaries, it is initially tagged as "both" and then returned to the Predictor-Corrector algorithm described in section 3.2.2. According to the results from that, we tag the word as **lang1**, **lang2** or **ambiguous**.
  - If it is not present in either of the dictionaries, we go through another list of processes described in section 4.2.

4.2 Non-dictionary words

If the word is not found in either the English or the Spanish dictionary, we use the following techniques:

- We check for the word's presence in the French, Spanish, Portuguese (Portugal), Portuguese (Brazil) and German dictionaries. If found, we label the word as a foreign word (**FW**).

[4]http://nlp.stanford.edu:8080/ner/

Tweet type	No. of tweets	F-score
Monolingual	6090	0.83
Code-switched	4626	0.75
Total	10716	**0.79**

Table 1: Tweet-level results

Label	Tokens	Precision	Recall	F-score
lang1	16944	0.509	0.449	**0.478**
lang2	77047	0.813	0.597	**0.689**
ambiguous	4	0.000	0.000	0.000
mixed	4	0.000	0.000	0.000
ne	2092	0.139	0.169	0.153
fw	19	0.000	0.158	0.010
other	25311	0.500	0.431	0.466
unk	25	0.002	0.240	0.003

Table 2: Word-level results

- If the word is still not found, we use the word-slicing algorithm spoken about in section 3.2.1 to see if it is a mixed word or not. If it is, we tag it as **mixed**.

- If the word is not mixed, we have failed to find any of the given criteria in order to fit, we label it as **unk** or unknown.

5 Results

Tables 1 and 2 summarize the tweet-level and word-level results for the test data, while the overall accuracy was determined to be **0.536**.

The accuracies are a bit lower than on the training and development data (where we achieved a best F-score of 0.772) and there are quite a few scopes for improvement that we can think of:

- The Named Entity Recognizer works based only on the English language. If we ran both English and Spanish NERs, that might have helped to improve the accuracy for "ne".

- For ambiguous words, our existing predictor-corrector method would lead to tagging more ambiguous words than there actually would be, since a lot of the surrounding words would be unknown/other. Moreover, those are simply tagged as ambiguous, while they are not indeed. We expanded our search area on finding non-tagged words, it may have helped increasing the accuracy here.

- For identifying the foreign words, we have considered the potential loss of accents while typing and that might have helped us to increase our detection for foreign words a bit more.

- In case of identifying the mixed words, we check the presence of word slices in dictionaries. However, many of these slices would be morphemes and not complete words and thus, wouldn't be found in a dictionary. We would need to develop a way of detecting presence of morphemes in a language for this. An N-gram pruning technique may help, but in code-switched contexts, with more than 2 labels to classify words in, it may not be as helpful as in a binary situation.

- Certain misspellings, typos, abbreviations and contractions may not have carried and been wrongly classified. We would need more sophisticated algorithms for detecting these cases.

6 Conclusion

We have achieved a healthy level of accuracy and utilized a fully developed algorithm without any machine learning or classifiers and also discover and discuss some areas of improvement and potential correction. In future works, we would perhaps tweak our algorithms in those ways to achieve a better accuracy.

References

René Appel and Pieter Muysken, 2005. *Code Switching and Code Mixing.* Amsterdam University Press.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics*, pages 363–370.

Bo Han, Paul Cook, and Timothy Baldwin. 2012. Automatically constructing a normalisation dictionary for microblogs. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 421–432, Jeju Island, Korea, July. Association for Computational Linguistics.

Fei Liu, Fuliang Weng, Bingqing Wang, and Yang Liu. 2011. Insertion, deletion, or substitution? normalizing text messages without pre-categorization nor supervision. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011)*, pages 71–76.

Fei Liu, Fuliang Weng, and Xiao Jiang. 2012. A broad-coverage normalization system for social media language. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, pages 1035–1044.

Stan Schroeder. 2010. Half of messages on twitter aren't in english [stats].

W3Techs. 2015. Usage of content languages for websites.