Code Switching Phenomenon in Thai Tweets

Ben P. Athiwaratkun
Statistical Science Department
Cornell University
pa338@cornell.edu

***Abstract*—Code switching is a phenomenon when there is a change in the language used in a sentence. This paper aims to investigate the Thai-English code-switching phenomenon that occurs in Thai tweets. We analyzed the distribution of code-switching English words compared to the word distribution for normal English usage. We also analyzed the Thai 1,2,3-grams that are correlated with the occurrence of code-switching with a logistic regression model. This model also gives us a predictor of the occurrence of code-switching.**

I. Introduction

A. Motivation

Code-switching can often be required when speakers talk about proper nouns such as movie name, person's name, etc. We refer to this as pseudo code-switching instances in this paper. Another type of code-switching is when there is an alternative word that is a translation to the original language of the code-switching word. In this case, there is no obvious need for the switch of languages. This phenomenon is quite intriguing and it will be the subject of our study. Below are such examples of what we consider true code-switching tweets involving the word `organize`.

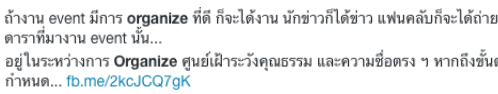


Figure 1: Example of Code-Switching Tweets

The translation of the two tweets are

- If the event is well organized, the fans will be able to take pictures with the celebrity, and the reporters will be able to get news reported.
- In the middle of organizing the Center of ...

The speakers' reason for choosing the word `organize` instead of an alternative word in the base language is not readily clear. Both of these tweets do not specifically require the English word `organize`. The hypotheses are that the choice to use English reflects the socio-economic status. It can also be the case that the English word `organize` is widely adopted such that speakers tend to use them interchangeably with the Thai word. Or it can be that the word `organize` help conveys the meaning better than the Thai alternative. This paper will investigate the structures of the code-switching instances. In particular, the questions we will attempt to answer are the following:

- What English words occur significantly more or less in code-switching instances compared to its normal English usage? Is there any clustering structure in terms of word similarity that we can infer from these words?
- What Thai words or phrases are significantly correlated with the occurrences of code-switch? Can we use these words to predict whether the author will code-switch in a Tweet?

B. Data Choice

Code-switching can be observed in many forms of communication methods, particularly in informal ones such as posts on social media platforms and forums. Twitter is a particularly suitable platform due to its convenient API. Note that the general methodology presented in this paper can also be extended for bigger dataset that contains not only Thai but other languages. However, for this paper, the dataset is limited to only tweets in Thai as data cleaning process heavily requires knowledge in the base language.

II. High Level Summary

Two major parts of this paper are:

(i) Analysis of code-switching words
- We look at the part of speech of code-switching English phrases and code-switching English unigrams as opposed to their normal English usage counterparts.
- We filter out and select only English unigrams that are non-proper nouns as true code-switching instances and compare them with normal English distribution, both individually and cluster-based.

(ii) Code-switching prediction
- We perform logistic regression with target variable being an indicator 'code-switching occurs' or 'no code-switching occurs'We use the occurrences of frequent Thai 1-,2-,3-grams as regression features.

III. Methodology

Figure 2 demonstrates a data preparation process for both (i) and (ii).

A. Filtering Duplicates

This phase ensures that there are no duplicate tweets in the data set. The data used are streaming data which contains very little duplicate tweets (same id), however, we find 55 tweets with duplicate `id_str` out of $2,583,847$ tweets.

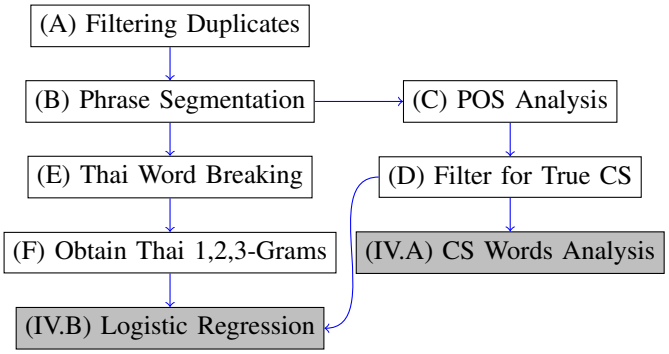


Figure 2: Data Pre-Processing Steps. The steps in shaded boxes are analyses of results which are explained in Section IV.

In addition to obvious duplicates, we filter out retweets. Note that the analysis of code-switching words without filtering retweets out is equivalent to treating a retweet as a regular tweet, which could be justifiable if the non code-switching tweets are as likely to be retweeted as the code-switching tweets. However, this assumption might not hold. Additionally, to perform regression, we need to filter out retweets in order to eliminate the possibility of prediction of seen data in the test set. We filter out tweets that contain the following pattern `r'RT @[\w]+:'` This filter is needed in addition to the `"retweeted":true` tag contained in tweet json format as manual inspection confirms that some tweets that are clearly retweets have this flag set as `"retweeted":false`. This can be the case that users manually retweets. The final dataset contains $775,400$ tweets.

B. Phrase Segmentation

We segmented tweets into a disjoint list of substrings. Each substring either contains only Thai characters (unicode range `[\u0E00-\u0E7F]`) or no Thai character at all. This preprocessing is needed since the Thai word-breaking library is not meant to break English word adjacent to Thai word with no space (which occurs quite often in tweet data). The library is also strictly incompatible with input containing unfamiliar unicode, especially ones that represent Emoji or symbols. The thai substrings are processed in (E) to obtain n-grams. The non-Thai strings are processed in (C) for part of speech analysis.

C. Part of Speech Preliminary Analysis

We define pseudo code-switching instances as the occurrences of English phrases that are adjacent to Thai phrases. These are not all what we consider true code-switching instances since words such as 'Harry Potter' or 'Interstellar' almost, if not always, necessitates the use of the English words.

The hypothesis is that these pseudo code-switching instances will contain a large number of proper nouns. In addition to proper noun category (denoted by ^), we also found marked difference in part of speech distributions, as shown in Figure 3. The part of speech tags are explained with examples in Table I

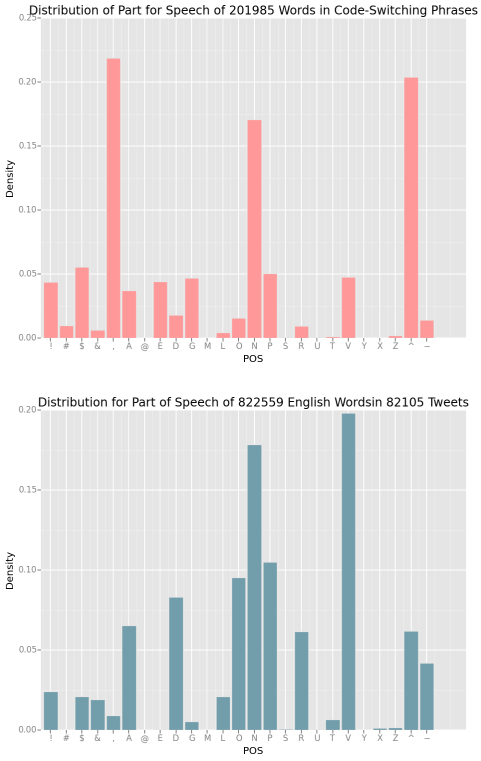


Figure 3: Distribution of Part of Speech for Words in Code Switching Phrases vs Words in English Tweets

We use the the observed ratio as an estimate for the probability (which corresponds to the MLE estimate). Figure 3 shows the sample distribution for part of speech for words that occur in pseudo code-switching English phrases, as well as the part of speech for English words in normal English tweets. There are $201,985$ words that occur in pseudo code-switching phrases. The baseline English tweets contain $822,559$ words.

Table I: Part of Speech Legend and Examples. See `[1]` for more details.

POS Tag	Tag Meaning	Examples
N	common noun	`mv line BTS mama cap winner`
O	pronoun	`me I Me II US i`
^	proper noun	`iPhone ELF Facebook 2NE1`
L	nominal + verbal	`ibaekrauhls ID iPhone6Plus`
V	Verb	`vs talk VS read VOTE Rewind`
A	Adjective	`infinite Favorite Fast`
R	Adverb	`Y forever here alone always`
!	Interjection	`gt http lt amp IG`
P	subordinate conjunction	`via by in Like`
&	coordinating conjunction	`n and or But`
T	verb particle	`up off down out`
#	hashtag	`#codeswitching`
@	at-mention	`@user`
~	discourse marker	`RT rt Rt PLSRT -rt`
U	URL or email address	`google.com/dtJfg04`
E	emoticon	`x XD T___T T__T -w-`
$	numeral	`2ndWin One 1stWin 10thirty`
,	punctuation	`yg_bear YG_iKONph i5 -v-`
G	other abbreviation	`SM iKON w M PlsRT`

In this paper, we will consider the code-switching unigrams

as true code-switching as it is generally challenging to determine if the code-switch is true for non unigrams. Figure 4 shows that we still keep a lot of data by choosing only the unigrams.

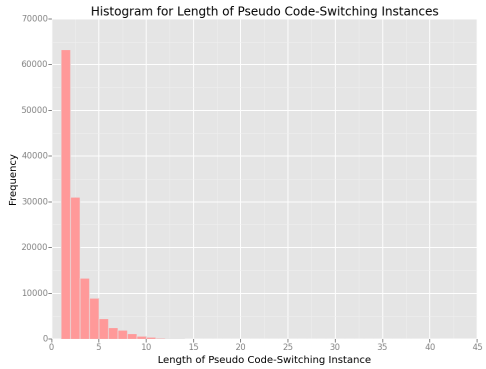


Figure 4: Histogram for number of words in pseudo code-switching phrases. There are $775,345$ pseudo code-switching phrases contained in $102,550$ Tweets.

Considering only the (pseudo) code-switching unigrams, the part of speech distribution is shown in Figure 5.

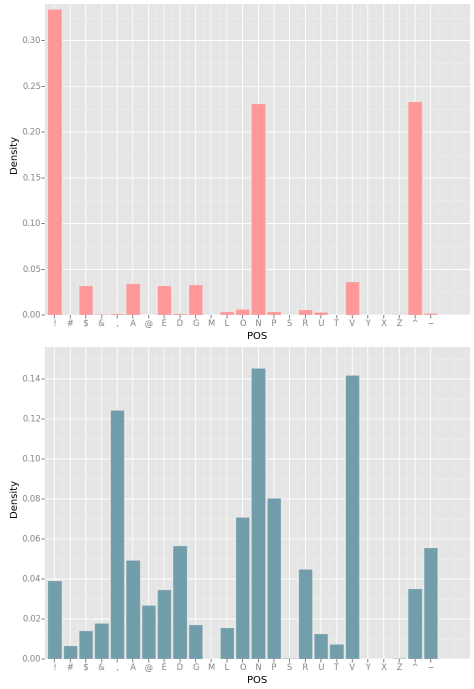


Figure 5: Distribution of Part of Speech for Words in Code Switching Unigrams vs Words in English Tweets

D. Filter for True Code-Switching Words

From this point on, we will attempt to come up with a filtering rule for true code-switching instances. As mentioned before, we will be considering only code-switching unigrams. In addition, the sample words from tagger I also suggest that we keep only the words with the following part of speech: `[&,A,O,N,P,R,T,V]` Note that ideally, we should be able to keep a larger set part of speech. However, based on the tags performed on a subsets of twitter words shows in Table I, we choose to filter out some potentially meaningful part of speech that ideally we should be able to use such as L (nominal + verbal).

In addition, there are some proper nouns that passed through the part of speech filter such as `interstellar, divergent, vine, galaxy`. This group contains proper nouns that might be recently popular and consequently is not incorporated in the POS tagger. We manually go through the code-switching words (this is possible due to its relatively small size of distinct words) to identify proper nouns and filter out words in the following list:

Table II: List of Proper Nouns for Manual Filter

Proper Noun	Explanation
'interstellar'	Movie name
'divergent'	Movie name
'insurgent'	Movie name
'kamikaze'	Music band
'line'	App name
'marvel'	Company name
'vine'	App name
'whiplash'	Movie name
'beam'	Singer name
'coke'	Company/Product name
'muggins'	Harry Potter term
'tot'	Company name
'galaxy'	Product name

E. Word Breaking (Thai)

The Thai phrases in process (B) are the input for word breaking. For a given Thai phrase or sentence, words are written contiguously without any space between them as seen in example tweets shown in Figure 1.

We use the library `libthai0.1.4` by Theppitak Karoonboonyanan et al. and a Python interface `PyThai` to break each Thai phrase into a list of words. The Thai word list will be further processed to obtain n-grams.

F. Obtaining N-Grams

We choose to obtain 1-,2-,3-grams of the words that have already been broken from process (E). Note that if we were to do n-gram over characters, this should be roughly $1-15$-grams for Thai. Note that due to our pre-processing step (B), we treat any white space, punctuation, and English character in the *original* text as a stopping character for the n-gram sliding window. This is appropriate for Thai because a space indicates the end a phrase or a sentence (unlike in English where white spaces indicates the end of a word). Punctuations are also usually not used in the middle of a Thai phrase and therefore their presence are presumed to indicate a switch to a new phrase, if any.

Figure 6 shows the histogram of Thai $1,2,3$-grams obtained from sliding window method on the lists of contiguous Thai words. As regression features, we keep only n-gram that occurs

at least 10 times. This reduces the number distinct n-grams from $3,739,286$ to $119,450$. The total number of n-grams collected is $17,538,739$.

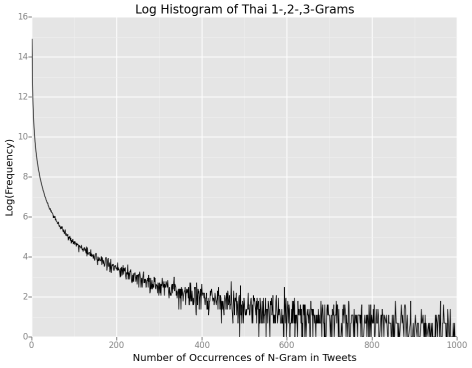


Figure 6: Histogram (Log Scale) of Thai 1-,2-,3-grams.

IV. Analysis and Results

A. Analysis of Code-Switching Unigrams

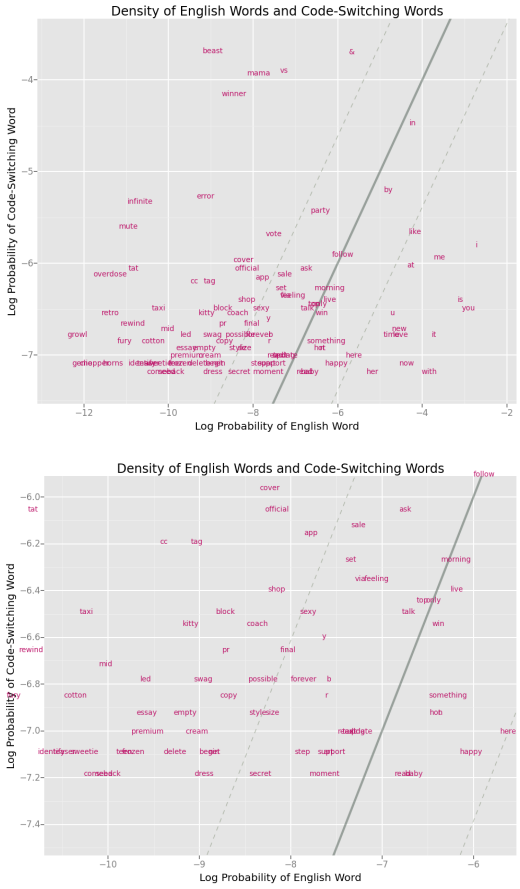


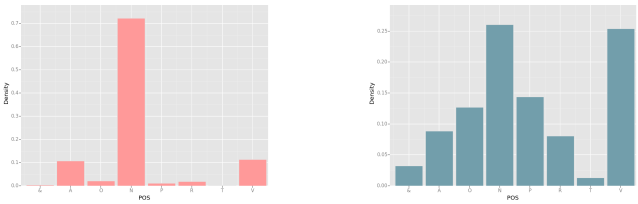
Figure 7: Distribution of part of speech for words in code switching unigrams versus words in English tweets. Dark line shows the line of equal probability. Dashed lines show the lines of probability ratio of 4.0 and 0.25.

We are interested in seeing how the code-switching language differs from the normal English language. In particular, we use the twitter data collected from 2008 to 2012 containing $56,345,753$ tweets as the baseline English language. This dataset is provided by CMU's Twitter NLP.

Figure 7 shows the words and their associated code-switching probabilities and English tweet probabilities. The probabilities are the observed ratio (or MLE estimate) given the group of $2,673$ distinct words that occur in both code-switching and English usage. The total number of code-switching words are $13,175$ and the total number of English words are $518,410,750$. On average, each word occurs roughly $\frac{13,175}{2,673} \approx 5$ times in code-switching. Figure 7 show the words that occur in code-switching at least 10 times.

The words in Figure 7 might appear to have no apparent structure. There are some words that, for a native speaker, seems to make sense that they are used quite often for code-switching such as party, vote, official, cotton, copy, shop, taxi, sale, mute, etc. Next, we attempt to cluster the code-switching words in order to infer some structural interpretation.

1) Part of Speech of True Code-Switching Unigrams: We also consider the distribution of part of speech but only for code-switching unigrams after filtering process. The baseline is the English unigrams filtered with the part of speech set `[&,A,O,N,P,R,T,V]` . Figure 5 shows the result.



(a) Distribution of POS for Code-Switching Words

(b) Distribution of POS for English Words

Figure 8: Distribution of Part of Speech for Words in Code Switching Unigrams vs Words in English Tweets

We note that where there is a somewhat equal balance between noun and verb for normal English usage, there is a disproportionately high number of nouns for code-switching. Adjective, however, is quite on part with verb for code-switching as opposed to normal English usage where adjective is less than half of the verb frequency. This might be because users tend to use words to help describe their feelings (adjective) rather and explain their actions (verb).

2) Brown Clustering: Next, we attempt to compare the code-switching words with English words using Brown cluster algorithm. The library is obtained from `https://github.com/percyliang/brown-cluster`

Figure 9 shows the density of each cluster for words that occur in both code-switching and English tweets. The sample words for each cluster is shown in Table III.

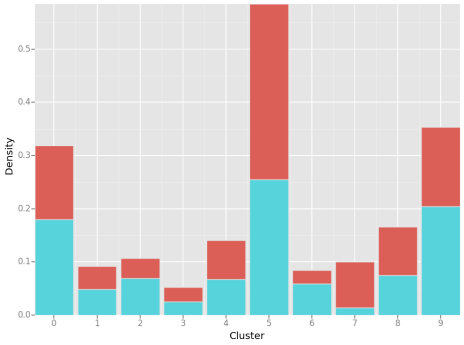


Figure 9: Cluster Density of Code-Switching Words and English Words. Red bars (top) indicates the code-switching densities. Blue bars (bottom) indicate English densities.

Table III: Sample Words for Brown Clusters. The words are shown from highest frequency or lowest.

Cluster	Sample Words
0	by, sms, follow, cc, feeling, live, sus, cotton
1	vs, sexy, you, sale, swag, art, new, social
2	at, tag, edit, m&m, again, alert, paper, cry
3	save, boxset, kitty, user, hashtag, depress, from
4	vote, ip, comeback, redcarpet, roommate, search, sad
5	mama, party, winner, error, inbox, me, beast, only, talk
6	overdose, empty, read, and, time, kpop, sound, portfolio
7	mv, sweetie, ep, final, public, local, verb, speed, original
8	in, me, retro, set, reading, text, begin, dvd, baby, as, story
9	w/, mute, cover, ask, via, shop, i, yu, possible, teen, essay

Note: even though there seem to be density disparities for some clusters such as cluster $5, 6, 7$, the sample words do not indicate that each cluster is easily interpretable. This might be because the number of clusters used is quite low and therefore words that might be seem to be directly related are forced to be in the same group. The original implementation which performs clustering with 1000 clusters seem to achieve good results in terms of cluster interpretability. However, we are quite constrained by the number of code-switching words since there are only ≈ 2000 distinct words. Figure 10 shows the cluster densities for 100 clusters. From observation, this is still too few of a cluster number to make the clusters easily interpretable. Sample words for cluster 48 is [party, like, official, support, morning, order] and [baby, sweet, account, said, country, 1st, colorful] for cluster 49. We did not perform statistical test for disparity of cluster densities due to lack of cluster interpretability.

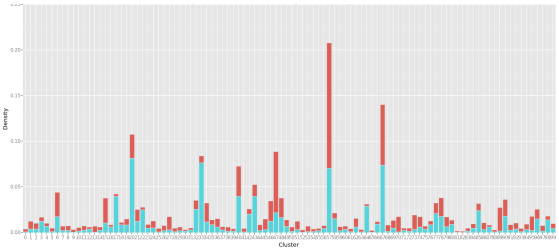


Figure 10: Cluster Density of Code-Switching Words and English Words. Red bars (top) indicates the code-switching densities. Blue bars (bottom) indicate English densities.

Note: In addition to brown word clustering, we also attempted a k-medoid cluster based on distance involving path similarity. However, for a large number of words, this algorithm turns out to create disproportionately big/small clusters. The algorithm is also quite slow compared to brown clustering and therefore we did not pursue this approach to the fullest extent.

B. Logistic Regression

In this section, we are interested in predicting when code-switching occurs in a tweet based on Thai n-grams. We choose to use logistic regression as a classifier model due to the interpretability each feature's influence (correlation coefficient β). Logistic regression is also quite efficient for very high dimensional feature space (order of magnitude $10^5 - 10^6$) and large data size.

The target variable for logistic regression is whether code-switching occurs in a tweet. The features are the occurrence of n-gram. The design matrix has $756,485$ number of observations and $119,450$ features.

Table IV: Average of prediction scores of 5-fold cross validations for both lasso and ridge regression. The predication score is defined by the percentage of correct prediction.

Regularization	λ	Prediction Score	Standard Deviation
Lasso	e^{-4}	0.972047	3.6637×10^{-4}
Lasso	e^{-3}	0.976628	3.6681×10^{-4}
Lasso	e^{-2}	0.980829	3.6053×10^{-4}
Lasso	e^{-1}	0.984475	1.5892×10^{-4}
Lasso	e^{0}	0.986524	7.2210×10^{-5}
Lasso	$\mathbf{e^{1}}$	**0.986922**	$\mathbf{9.77314 \times 10^{-5}}$
Lasso	e^{2}	0.986816	2.15595×10^{-5}
Lasso	e^{3}	0.986745	3.14490×10^{-5}
Lasso	e^{4}	0.986494	7.93141×10^{-6}
Ridge	e^{-4}	0.979649	3.08827×10^{-4}
Ridge	e^{-3}	0.982186	2.71405×10^{-4}
Ridge	e^{-2}	0.984274	1.56487×10^{-4}
Ridge	e^{-1}	0.985726	1.23270×10^{-4}
Ridge	e^{0}	0.986445	9.32293×10^{-5}
Ridge	e^{1}	0.986806	4.93549×10^{-5}
Ridge	$\mathbf{e^{2}}$	**0.986917**	$\mathbf{4.79543 \times 10^{-5}}$
Ridge	e^{3}	0.986903	3.95688×10^{-5}
Ridge	e^{4}	0.986524	2.18813×10^{-5}

First, we perform a regularized logistic regression with 5-fold cross-validation to determine the optimal regularization parameter λ. The penalty term considered are both ℓ_1 norm

(lasso model) and ℓ_2 norm (ridge regression model) of the coefficient vector $\vec{\beta}$. We use the library `scikitlearn` with the design matrix X in `Python`'s `scipy` sparse matrix format.

Table V: Prediction scores for the baseline estimator (predict all non code-switch) and the classifier trained by regularized logistic regression. The scores are based on 20% of data, which contains $151,297$ samples with $2,098$ code-switching instances.

Estimator	Accuracy	Precision	Recall	CS Pred.
Flat	0.98613	N/A	0.0	0.0
Lasso $\lambda = e^1$	0.98683	0.7977	0.068	178
Lasso $\lambda = e^0$	0.987587	0.85484	0.126	310
Lasso $\lambda = e^{-1}$	0.990912	0.95471	0.361	794

The optimal λ based on the prediction score for Lasso model ($\lambda = e^1$) leads to a high-bias estimator that predict predominantly non code-switch. If we allow for more variance which corresponds to the assumption that irregularity/code-switching in data are meaningful signals, we obtain a classifier that tends to predict more code-switching as shown for the case $\lambda = e^0, e^{-1}$. In the sparse-data situation, we might prefer the high-variance lower-bias predictors since the low-variance one simply needs more data in order to be confident about predicting code-switch. For the Lasso model with $\lambda = e^{-1}$, we obtain the classifier that high relatively high recall of 0.361 which means that out of all code-switching instances, it can identify 36% of them. This is reasonably high because given the right features positively correlated to code-switching, the phenomenon occurs with some probability as some users might simply choose to use Thai words instead of code-switching to English. Out of all the identified code-switching, however, 99.09% are correct.

Using the Lasso model with $\lambda = e^1$, we perform logistic regression on the entire dataset and analyze the significant features. Out of $119,450$ features, $8,198$ features are found to be significant at level $\alpha = 0.001$. The number of positive β that are significant are $1,918$.

Note: please see analysis for significant Thai n-grams with example tweets in separate Appendix. (due to difficulty of Thai with LaTeX)

V. Conclusion

We succeed in determining code-switching words at a satisfactory level. From this we build a classifier which is able to predict code-switching instances better than the baseline flat predictor and analyze significant n-grams that are correlated to code-switching. While there is no immediate interpretation for the significant n-grams, we get a glimpse into what types of n-grams influence code-switching. These types of n-grams tend to be more modern activity such as n-grams with meaning button, apps, concert and some other non-obvious but perhaps more interesting n-grams such as job/event, city, clothes.

VI. Discussion

This section discusses the challenges the further improvement that can be made.

A. Code-Switching Determination

One of difficult tasks in this research is the determination of whether a code-switch occurs in given tweet. The process employed filters out proper noun quite well to some extent. However, we find that there are some words that the POS tagger could not have known. An example of this is the following tweet.

Figure 11: Example Tweet

The word `Line` in this case refers to a messaging application that is particularly popular in South East Asia. In this paper, we manually goes through the words and select out the set of words that are most likely proper nouns such as `line, interstellar`, to name but a few. However, it could very well be the case that the tweet actually refers to the word `line` in the usual English meaning. An improvement for the code-switching determination process would be a classifier that translates the whole phrase into English and performs the tagging.

B. Sparsity of Code Switching Data

In this exercise, the code-switching tweets occurs with frequency roughly 2%. This translates to about 13K code-switching instances with 2.6K distinct words. This is quite small compared to the number of words in the English baseline of 800K distinct words. Given more time and resources, this project can be extended to a larger data set. If we gather enough data such that the number of distinct code-switching words is close to the order of $10 - 100$K, this will allow for more rigorous analysis on the comparison of code-switching and English word clusters. A natural extension would also be to categorize the words against LIWC. This might also need more code-switching data in order to obtain the significance (if any) as well.

Acknowledgment

The author would like to express sincere gratitude for the following NLP-related libraries that help this project implementation manageable/possible.

- NLTK
- Python's langid
- Python's enchant
- ark-tweet-nlp-0.3.2
- ark-CMU Brown cluster implementation
- scikitlearn

References

[1] K. Gimple, N. Schneider, B. O'Conner et at., *Part-of-Speech Tagger for Twitter: Annotation, Features, and Experiments*

[2] P. Liang. *Semi-Supervised Learning for Natural Language*, Department of Electrical Engineering and Computer Science, MIT, 2005. `https://github.com/percyliang/brown-cluster`