Identification and Analysis of Conversational Codeswitching Triggers

Technical Report

September 4, 2018

Student Co-PIs:

Jocelyn Alvarado
joalvar1@student.uiwtx.edu
Department of Mathematics
University of the Incarnate Word
Advisor: Dr. Theresa Martines

Dr. Chen Chen
carachenchen@gmail.com
Department of Earth, Atmospheric, and Planetary Sciences
Purdue University

Rouzbeh Shirvani
rouzbeh.asghari@gmail.com
Department of Electrical Engineering and Computer Science
Howard University
Advisor: Dr. Mohamed Chouikha

Dr. Yanina Shkel
yshkel@princeton.edu
Department of Electrical Engineering
Princeton University

Taylor Williams
taylorhallwilliams@gmail.com
Department of Bioengineering
University of California, San Diego
Advisor: Dr. Todd Coleman

1 Introduction

In language, codeswitching occurs when a speaker uses two or more languages in the context of one conversation. Speculation on motivation for switching is multifaceted; it is possible that a person may switch languages to hide certain information from listening native speakers, to better express themselves because certain words cease to exist in a given language, or to accommodate the person with whom they are speaking. Very few scholars have investigated prediction methods related to codeswitching [2, 3]. Though it has been proposed that conversation dynamics affect factors of codeswitching, even fewer articles exist on the mathematical quantification of the hypothesized correlations between two individuals' switching frequency [6].

In this project, the team aimed to better understand the mechanisms of codeswitching by using Natural Language Processing (NLP) techniques. In particular, the team was interested in investigating and developing algorithms to predict when codeswitching will occur based on several conversation features. The team had access to an expert-annotated Swahili-English data set during the Center for Science of Information (CSoI) data science workshop at Purdue, as well as throughout the year. The preliminary analysis was done during the CSoI workshop using R for basic data manipulation. The team carried out a frequency analysis of codeswitching triggers and used data visualization to reveal the dynamics of codeswitching on a conversational level. The preliminary results suggested that certain words have better predictive value for codeswtiching. Moreover, there was a clear empirically observed correlation between the codeswitching behavior of the two speakers in a conversation.

The team built on these initial findings and applied *supervised*, as well as *unsupervised*, learning algorithms to the codeswitching data. First, the team focused on *classification*: a supervised learning task where the goal is to learn to accurately classify a data point's label given its set of features. The first problem that the team encountered is that most classification algorithms use features which are numerical vectors, while many of the important features in the codeswitching data (e.g. words, parts of speech, language spoken) are categorical. The NLP technique used to tackle this problem is known as *word embedding*, and the team investigated a number of word embedding techniques. The team focused on the simplest method called *one-hot encoding* and implemented it together with a *Naive Bayes* classification algorithm for prediction of codeswitching at an *utterance level*. That is, the predictor used the features associated with an utterance of speaker A to predict if speaker B's response will contain a codeswitch. The Naive Bayes is a good baseline algorithm because it is simple and known to work well for NLP tasks. The obtained results demonstrate that the features associated with an utterance of speaker A do indeed have predictive value for whether speaker B will codeswitch, and the Naive Bayes algorithm has good performance. Secondly, the team looked at unsupervised learning task and applied *topic modeling* techniques to the codeswitching data. The team found that there is significant overlap between most frequent codeswitching word and the topic of the conversation. The team used python and SKLEARN [7] – a scientific library that has different supervised and unsupervised machine learning algorithms – for this analysis.

The rest of this report is structured as follows. An overview of the preliminary work done at the CSoI workshop in May 2017 is given in Section 2. Possible approaches to constructing feature vectors using word embeddings are described in Section 3 and the approach of using the Naive Bayes classifier for prediction is described in Section 4. The topic modeling analysis of frequent codeswitching triggers is presented in Section 5. The report is concluded in Section 6.

2 Preliminary Results

To understand codeswitching triggers and the conversation dynamics, the team worked primarily with a dataset of 30 interview scripts consisting of conversations between bilingual Swahili and English speakers. The dataset was manually tagged to signify Swahili phrases, as shown in the example phrases below.

> **Original:** Okay. *<Swahili> Na unafikiria n </Swahili>* important *<Swahili> kujua </Swahili>* native language?
>
> **Translation:** Okay. *<Swahili> and do you think it is </Swahili>* important *<Swahili> to know </Swahili>* native language?

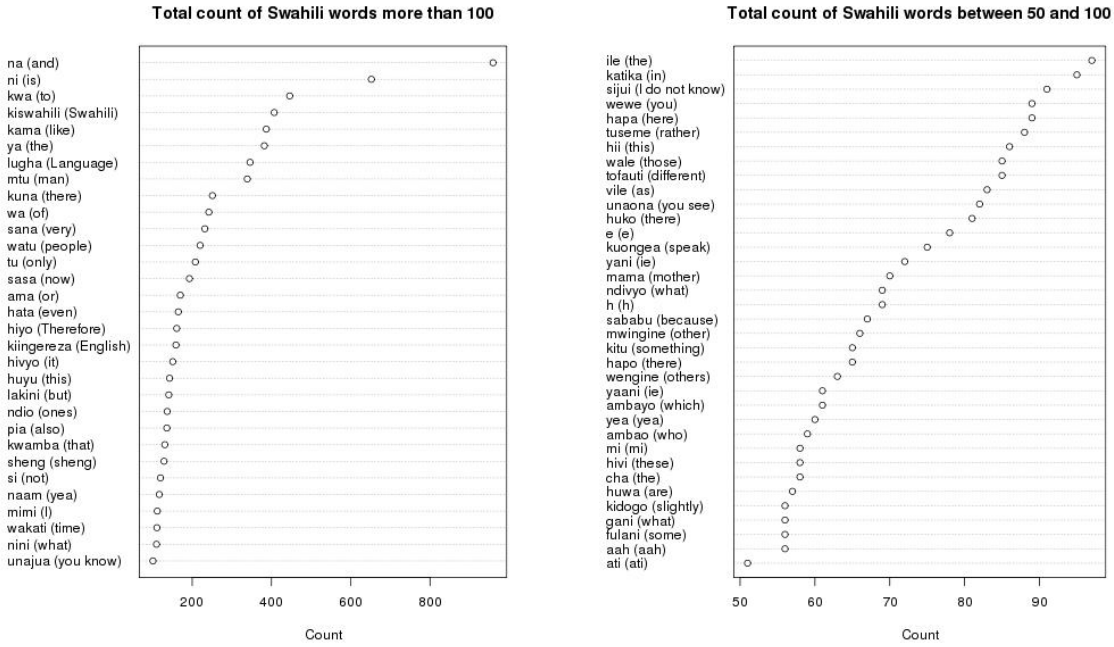

Figure 1: Swahili word counts ranked by their frequency

The team completed preliminary analysis in R, using regular expression functions, such as grep, gsub, and gregexp, and elements from the string package, stringr, such as str_match_all, str_count, and strsplit to extract several pieces of information. The translation from Swahili to English was done using Google Translate.

First, the count of each Swahili word was gathered. High frequency words, which indicate popular words in Swahili were ranked (Figure 1). The team identified the following topics that were frequently covered in the interviews: language, family, time and place. Second, the team was interested in learning what the words at switch points are, so the words before and after a switch happened were extracted from the conversations and analyzed. We ranked the words based on their frequency (Figure 2). Most of the words are filler words, in both English and Swahili. Words related to language, such as 'English', 'Swahili', 'language', 'accent' were frequently mentioned when switching to a different language.

The team also explored visualization of conversation dynamics, as shown in Figure 3. From this, the team hypothesized that interviewer switching frequency is correlated with interviewee switching frequency. Consider Figure 4 where the same speaker (Andrew) exhibits different switching behavior depending on the interview. This could be, for example, because the speakers influence each other's codeswitching frequency, or because the topic of the conversation could impact the codeswtiching frequency. On the other hand, the team observed that the speaking habits of the particular speaker also play a role in codeswitching. Consider Figure 5 where a low frequency switching speaker (Dustin) displays the same codeswitching frequency across two different interviews. At the same time, a high frequency switching speaker (Fred) also displays the same frequency across two different interviews. These preliminary observations served as a jumping off point for exploring switching prediction methods.

3 Word Embeddings

Investigating natural language processing from a predictive or causal perspective requires significant computational power to analyze the large and complex data that results from human conversation. The team encountered this when exploring conversation dynamics, namely those related to interviewee/interviewer switching frequency based on words spoken at switch points. Prior to classification or prediction, the dataset first needed to be converted from qualitative features to numerical data. We found that dimension reduction may then be necessary to best

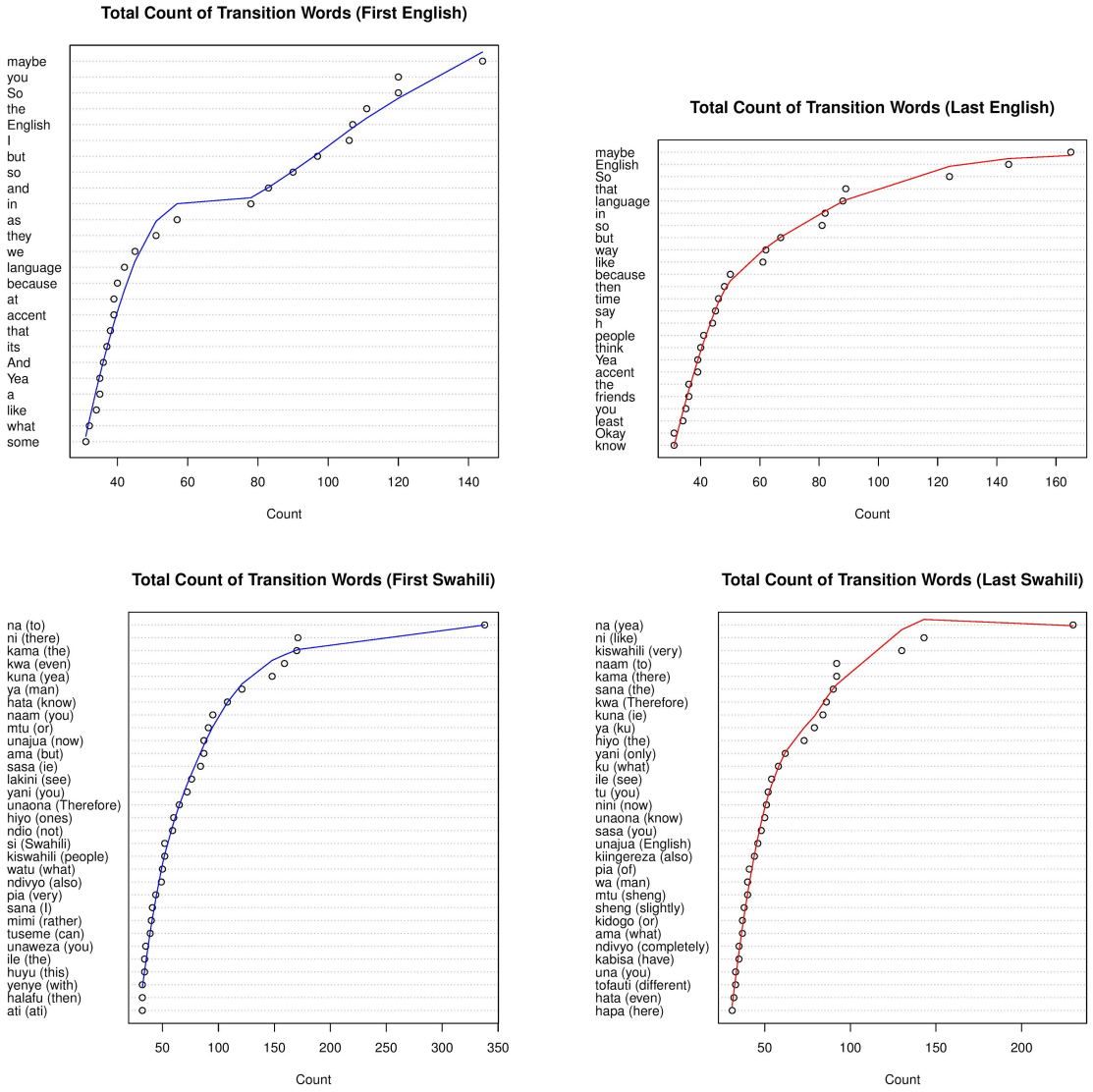

Figure 2: Word count of switch words. Top: Swahili switch words; bottom: English switch words.

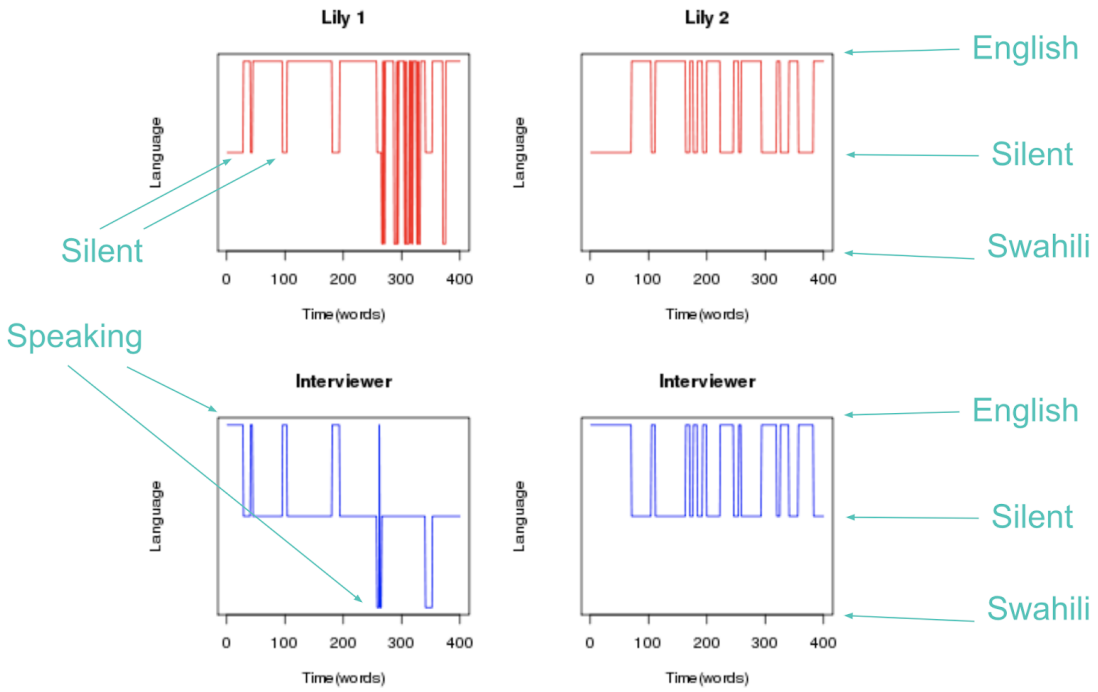

Figure 3: Conversation dynamics during an example interview.

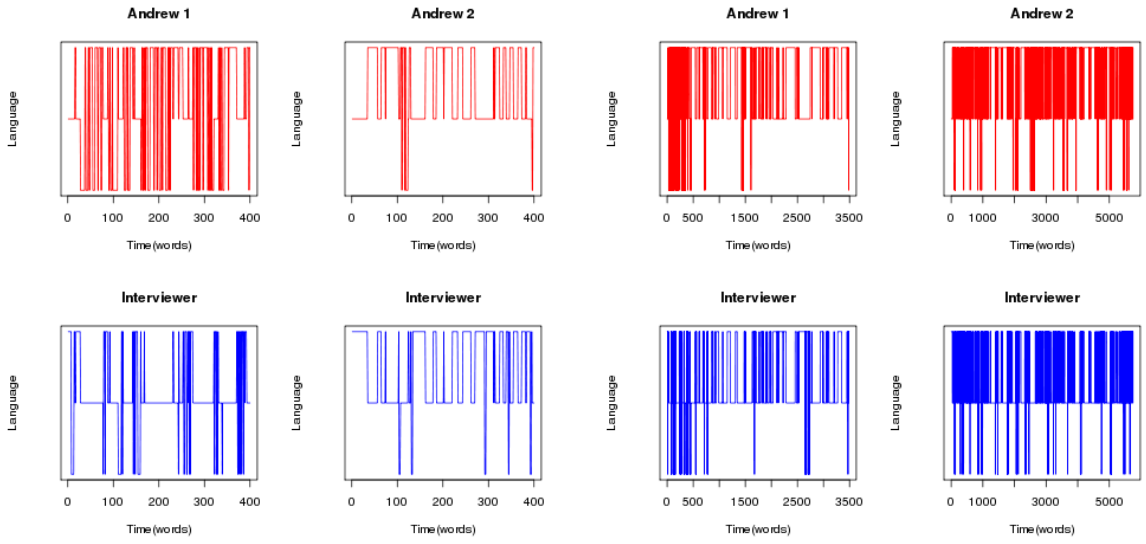

Figure 4:

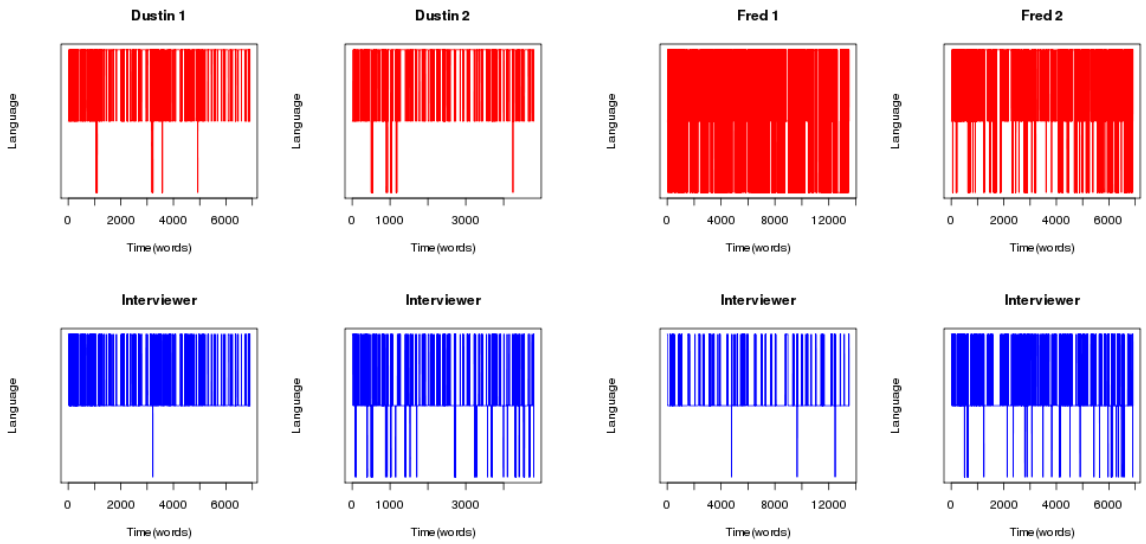

Figure 5:

analyze the data while improving computational efficiency and still preserving the meaningful information within the dataset.

To approach the idea of dimension reduction, the team chose to probe the technique of clustering. By clumping data into categories, prediction algorithms that use clustering can perform better because the datasets they operate on are often computationally easier to handle. In natural language processing, word embedding serves as a viable option for clustering because it maps words to numerical vector space [10]. This numerical data assists algorithms with learning and prediction by restructuring the data to a user-specified dimension, n, equal to the amount of desired categories. Figure 6 demonstrates this mathematical mapping of words to numerical vector space. The numerical array produced for each word is of size $n = 64$. We used Principal Component Analysis (PCA) to reframe the data into two-dimensional Cartesian coordinates for ease of comprehension. In this example, we extracted the words from Polyglot, which is an online database of word embeddings for over 100 languages [1]. The Polyglot word embeddings were created by training corresponding Wikipedia datasets up to 100,000 words large, primarily on their part of speech.

After using word embeddings to convert the conversational data from qualitative to quantitative, we used the K-Nearest Neighbors (KNN) learning algorithm to test the Polyglot word embeddings technique. KNN numerically calculates the closest k elements to the input element, given their mathematical distance in the n-dimensional space [5]. The team chose to use the word 'Maybe' as the test input to KNN because in our preliminary analysis discussed in Section 2, we found that the English word was used most frequently at codeswitch points. With $k = 10$, the KNN algorithm returned a vector containing the following words: 'Maybe', 'Perhaps', 'Thats', 'Ideally', 'Possibly', 'Hopefully', 'Surely', 'Unfortunately', 'Preferably', and 'Luckily'. Though no further mathematical analysis was performed, we found these preliminary results confirmed the validity of the Polyglot mapping technique at a high level.

Although advanced word embeddings like Polyglot coupled with an algorithm like KNN can be useful in terms of dimension reduction and information preservation, the team chose to use the simplest embedding technique, one-hot encoding, to map the dataset of words to numerical vector space. One-hot encoding preserves all of the information in a conversational or categorical dataset because it maintains the original dimension of the data. This mapping technique converts each word from an n-dimensional array of strings to an $nx1$ array that contains one unique element that is set to 'high' or '1' while all other elements are set to 'low' or '0'. The rows are then concatenated to form an nxn matrix where each word is represented by a unique index where the value is 'high'. An example of this is shown in Figure 7 [4]. By using one-hot encoding, we converted our conversational data to a binary dataset that classifiers and other prediction algorithms can most easily handle.

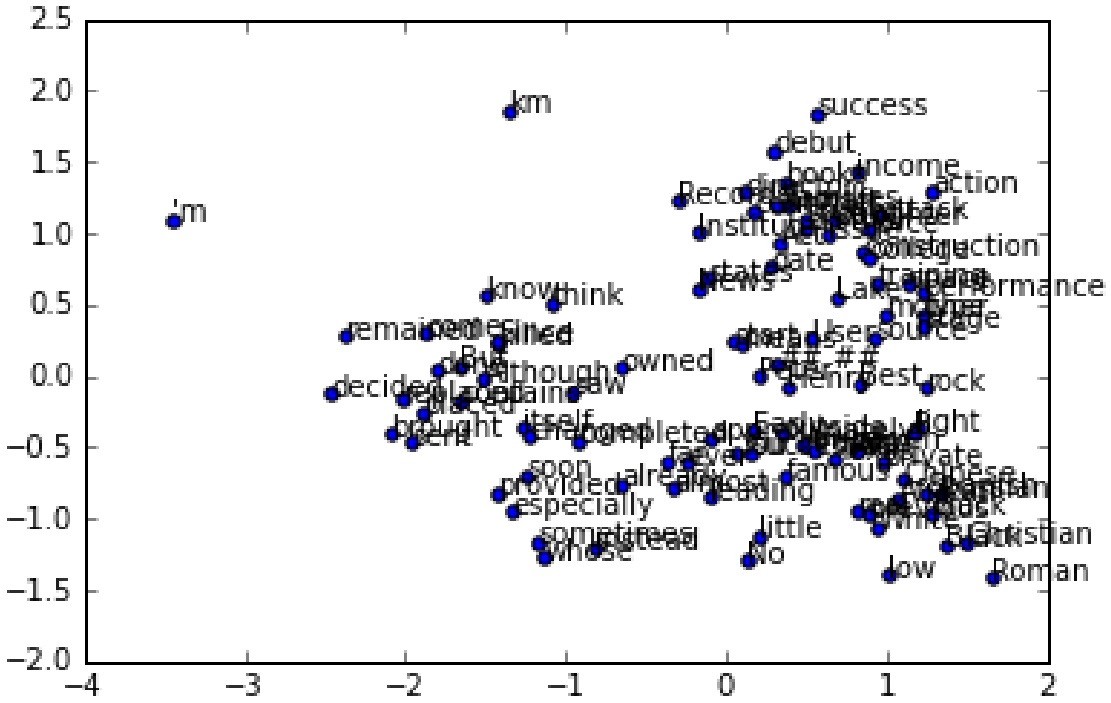

Figure 6: Vector representation of words from data index $i = 700 : 800$ in the $n = 100,000$-word English dataset, found using Polyglot word embedding approach. PCA was performed to view the data in two-dimensional space.

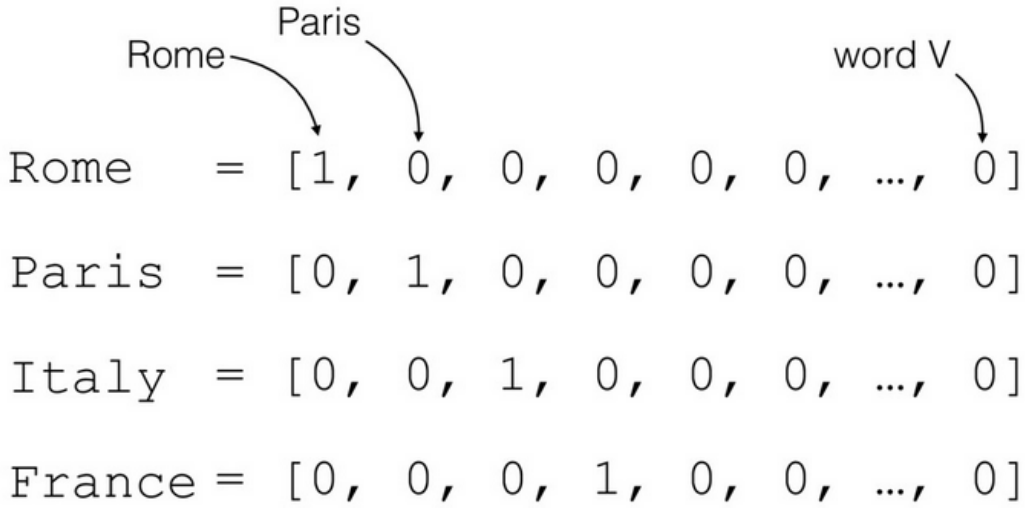

Figure 7: An example of one-hot encoding applied to words where one element is set to 'high' and all others are set to 'low'. The concatenated arrays would results in the one-hot encoded matrix, used as an input for classifiers and other learning algorithms.

4 Naive Bayes

The team implemented an utterance-level codeswitching predictor using a Naive Bayes classifier. Naive Bayes classifier is a simple classifier that is widely used in NLP tasks; for example, it has been particularly successful in tasks like spam detection and subject classification. The advantages of the Naive Bayes classifier is that it is easy to implement, and it scales well with the number of features. Its disadvantage is that it makes a very strong assumption that, conditioned on the label, the data features are independent; this is almost certainly false in most applications. Nevertheless, the classifier ends up working well in many practical problems [7].

4.1 Overview

Naive Bayes is a family of supervised learning algorithms used for classification. The aim of classification is to learn to predict a label y given a set of features $(x_1, \ldots, x_n)$. This can be done by first constructing a probability model; that is, estimating the probability of the label given the features. Using Bayes theorem we can write

$$P_{Y|X_1,\ldots X_n}(y|x_1, \ldots x_n) = \frac{P_{X_1,\ldots X_n|Y}(x_1, \ldots x_n|y)P_Y(y)}{P_{X_1,\ldots X_n}(x_1, \ldots x_n)}. \tag{1}$$

The Naive Bayes classifier works by assuming that the probability of each feature is independent given the label. That is,

$$P_{X_1,\ldots X_n|Y}(x_1, \ldots x_n|y) = \prod_{i=1}^{n} P_{X_i|Y}(x_i|y). \tag{2}$$

Thus, it works by estimating each $P_{X_i|Y}(x_i|y)$, as well as the class prior $P_Y(y)$, independently. Given these estimates it is then possible to estimate the probability of a label given a set of features

$$P_{Y|X_1,\ldots X_n}(y|x_1, \ldots x_n) = \frac{1}{Z} P_Y(y) \prod_{i=1}^{n} P_{X_i|Y}(x_i|y) \tag{3}$$

where $\frac{1}{Z}$ is a normalizing constant used to make the above a probability distribution. Note that this gives a probability distribution over possible labels. In order to get a hard decision as to which class to predict, a maximum a posteriori (MAP) decision rule is used:

$$\hat{y} = \arg\max_{y \in \mathcal{Y}} P_Y(y) \prod_{i=1}^{n} P_{X_i|Y}(x_i|y). \tag{4}$$

Here $\mathcal{Y}$ denotes the set of classes we are trying to predict; in the present case $\mathcal{Y} = \{0, 1\}$. Note that it is not necessary to compute $P_{X_1,\ldots X_n}(x_1, \ldots x_n)$ in order to make the decision, since it will be the same for all labels. It is worth noting that Naive Bayes is known to be a good classifier, but a bad estimator [7].

There are different versions of Naive Bayes classifiers, depending on the form of $P_{X_i|Y}(x_i|y)$. For example, SKLEARN [7] implements Multinomial, Bernoulli, and Gaussian Naive Bayes classifiers. Moreover, SKLEARN Naive Bayes uses *Laplace Smoothing* which is a technique used to mitigate the impact of infrequently observed features during the training phase.

4.2 Naive Bayes for Codeswitching

The Swahili-English data set contains 30 interviews. Each interview is a conversation between two parties that consists of a series of *utterances*. The number of utterances in each interview varies from 79 to 842. Within each interview the utterances alternate between the two speakers. That is, the first utterance belongs to the interviewer, the second to the interviewee, the third to the interviewer, and so on. The speaker of the utterance is not explicitly marked in the dataset, butthe two speakers always alternate. Each utterance consists of words and each word is labeled with word features such as part of speech, word language, whether the given word is a codeswitch point, etc. The current analysis focuses on two features: the value of the word and whether this word is a codeswitch point.

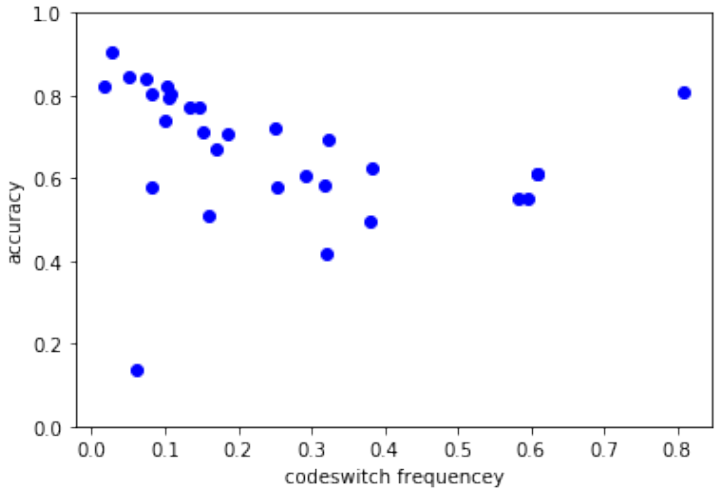

Figure 8: Accuracy measures the fraction of predictions which were correct. Although the plot is noisy, it does show that the features do have good predictive value for the label. In particular, it is not very surprising that for interviews with low codeswitching frequency the accuracy is high. Indeed, a simple classifier that always predicts no codeswtich would get high accuracy for those interviews. However, looking at the interview on the right side of the plot, the one with 80% codeswtich frequency, we see that it has good accuracy. Since in this case the test set has far higher codeswitching frequency than the training set, the only way the classifier can detect this is through the features.

The team's first task was to construct a (label, feature) vector of utterances. There are 10011 utterances in this data set, and the team represented each utterance as a vector $(y, x_0, x_1, \ldots, x_n)$. The label, y, is binary valued and denotes if a codeswitch occurred in the *next* utterance. In other words, given an utterance, the goal is to predict if the other speaker in the conversation switched languages when replying to it. The feature x_0 is also be binary valued and denotes if a codeswtich occurred in the *current* utterance. The remaining features $x_1, \ldots, x_n$ denote which words were used in the utterance, and how many times. For example, suppose that the kth word in the dictionary is 'you'. Then, $x_k = 3$ means that the word 'you' was used three times in the given utterance.

The dataset has about 17% of positive labels. That is, 17% of the time, an utterance contains a codeswitch. A typical run of the Naive Bayes classifier gives a *Precision* score of about 37%, a *Recall* score of about 28%, and an *F1* score of about 32%. This means that the classifier identifies about 28% of codeswitches correctly, and when predicts a codeswitch it is correct 37% of the time. This may not seem great, but considering how rare codeswitching events are this is actually good performance. Consider, for example, a classifier that just guesses that a codeswitch will happen with probability 0.17. Such a classifier will have Precision/Recall/F1 score of 0.17. The Naive Bayes classifier is doing much better than this; In other words, the features that it is using do have useful predictive value. Next, the classifier was rerun on the training data and similar performance benchmarks were obtained. This indicates that the classifier is not *overfitting* the data.

The team also looked into how our well our classifier predicts codeswitches within each interview. The team used *leave-one-out cross validation* for this. In other words, one fixed interview was put into the test set, and the remainder of the interviews were put into the training set. This was repeated for every interview. The results of this analysis are plotted in Figures 8, 9 and 10. The codeswitching prediction appears to be easier when the codeswitching frequency is higher. The Naive Bayes predictor is good, however, there is a lot of room for improvement.

5 Topic Modeling

The team observed an interesting pattern during the exploration of codeswitching data: Words that occur at the codeswitching points seem to be relevant to the topic of the conversation. Except for work presented in [9], there appear to be no studies that look into the relationship between

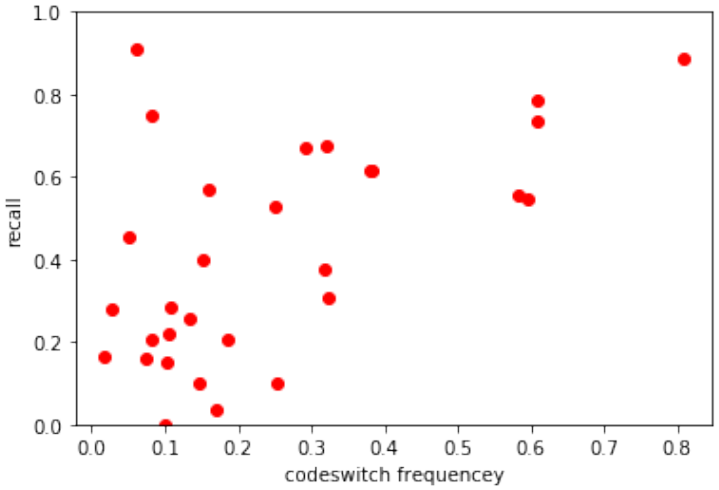

Figure 9: Recall measures the fraction of codeswitches that were identified correctly by the classifier. Note, that although the classifier does very well on some low-codeswitch-frequency interviews, it tends to do better when the frequency of the codeswitch is high.

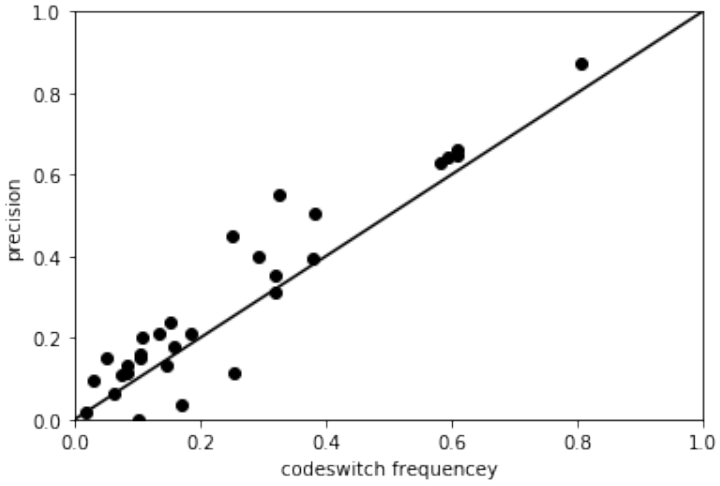

Figure 10: Observe that the classifier performs better when the frequency of the codeswitch is high. For precision we would expect that a simple probabilistic classifier (that randomly decides between switch/no switch while ignoring the features) to have the performance on the black line plotter. Although the plot is noisy, the Naive Bayes classifier outperforms this simple classifier and this again suggests that the selected features have a predictive value.

Non-negative Matrix Factorization	
Topic 1	swahili speaking really luo friends languages campus example laughs school
Topic 2	lugha kiingereza kiswahili naam tofauti kuna wale tuseme ama sijui
Topic 3	kiswahili french laugh_token ve speaking friends kikuyu languages school class
Topic 4	kipsigis french kind vocabulary really laughs words friends mean depends
Topic 5	mtu kuna ama kiswahili ndio yani unajua huyu hapa agree
Topic 6	actually tend communicate comes friends french important languages accent campus
Topic 7	interrupted agree luo tongue mother lot really hard speaking kikuyu
Topic 8	naam kikuyu kuna tofauti tuseme understand ndio friends native different
Topic 9	unaona swahili laugh_token french ndio cuz ve si accent tofauti
Topic 10	cuz luo friends words laughs laugh_token sijui french family class

Table 1: Topic Modeling based on Non-negative Matrix Factorization

codeswitching and topic modeling. In this section, we use SKLEARN [7] tool for topic modeling analysis related to codeswitching triggers.

5.1 Overview

Topic modeling is a statistical approach used in NLP to extract abstract topic labels from a set of documents [8]. When a document could be described by more than one topic, topic modeling can take that into account by considering more than one cluster of similar words. Topic modeling could be useful for information retrieval application when there are thousands of digital documents and we are looking for some specific topics among these documents. Going through these documents one by one could be tedious and time consuming and topic modeling allows to automate this process. Topic modeling is an example of an *unsupervised learning* task since the algorithm is not given training data, but needs to learn the topic from data.

5.2 Topic Modeling for Codeswitching

First, the team took a look at the words that have been most frequently used at the English or Swahili switch points. In the Swahili-English dataset the top switch words at English switch points are: 'Maybe', 'English', 'Accent', 'Language', 'Campus', 'Home', 'Story', 'Because', 'Lecturer', 'Influence', 'Friends', 'Place', 'Mostly', 'Time', 'Hard', 'Sometimes', 'Party', 'Come', 'Level', 'Change', 'Formality', 'Town', 'Actually', 'Feel', 'Get', 'Express', 'Words', 'Culture', 'Importance', and 'Difference'. The top switch words at Swahili switch points are: 'Ya', 'Na', 'Ni', 'Kuna', 'Yani', 'Kama', 'Kwa', 'Unajua', 'Unaona', 'Ama', 'Hiyo', 'Ile', 'Ati', 'Sana', 'Una', 'Lakini', 'Ku', 'Kujua', 'Kabisa', 'Hata', 'Kidogo', 'Unapata', 'Ina', 'Nini', 'Sijui', 'Iko', 'Hapa', 'Pia', 'Ana', 'Yake', 'Sasa', 'Wa', 'Za', 'Mimi', and 'Ninii'.

The team used topic-modeling tools in SKLEARN to extract the top 10 topics of the interview data. Tables 1, 2 show results of running topic modeling on the whole interview data and extracting 10 topics. These tables are based on Non-negative Matrix Factorization and Latent Dirichlet Allocation, respectively. It is readily apparent that there are similarities between the results of topic modeling and frequent codeswitch words both of which give some insight about the topic of the conversation. For example, words like "Language", "Accent", and "English" occurred in both topic modeling analysis and switch points. This phenomena is not totally unexpected, since the participants are talking about the choice of language at home/school in their conversation, it is reasonable to expect switch words be about the same subjects like language, school, English, and Swahili. As one can see, there are a lot of commonalities between the results of these two Tables 1, 2 and most frequent words at the switch points.

6 Concluding Remarks

In this project, the team learned about word embedding techniques for NLP tasks, implemented a codeswitching predictor at an utterance level using Naive Bayes classifier, and conducted

Latent Dirichlet Allocation	
Topic 1	mtu kiswahili swahili kuna ama unaona huyu ndio hapa agree
Topic 2	kipsigis lugha really kiswahili swahili level different words possible mother
Topic 3	swahili friends yani mtu languages kiswahili tend speaking campus important
Topic 4	kiswahili swahili really mtu ama kipsigis words kuna vocabulary huyu
Topic 5	kiswahili lugha kuna speaking friends languages school kiingereza example mtu
Topic 6	kiswahili actually french important friends doing things different luo tongue
Topic 7	kipsigis kind friends kiswahili french ve mix vocabulary tend come
Topic 8	swahili words kiswahili different mtu vocabulary really speaking friends place
Topic 9	kiswahili luo speaking better swahili agree interrupted mix tongue mother
Topic 10	luo swahili really mother tongue speaking friends interrupted agree lot

Table 2: Topic Modeling Based on Latent Dirichlet Allocation.

a topic modeling analysis of codeswitching point. The team used expert annotated Swahili-English interview dataset, and SKLEARN – a python machine learning library.

For the Naive Bayes classification the team used words contained in an utterance and if the current utterance contains a switch point as features to predict if the next utterance will contain a switch point. The presence/absence of a switch point is a reasonable feature; As observed in the preliminary work, the frequency of switching of one speaker in the conversation is often correlated with the frequency of switching of the other speaker. Moreover, the words contained in the utterance are related to the grammatical structure of the utterance, as well as to the topic of the conversation. Thus, they could reasonably predict codeswitching too. The team is currently investigating other features that could be useful for prediction. For example, using current codeswitch together with the speaker's identity, and parts of speech could be useful features.

In addition, the team is considering using information theoretic model selection techniques for feature selection [11]. The team is also considering repeating this analysis with more data [12], as well as more sophisticated machine learning techniques. In the later case, the question of word embedding for bilingual data would also need to be addressed in more detail.

References

[1] Al-Rfou, Rami and Perozzi, Bryan and Skiena, Steven. Polyglot: Distributed Word Representations for Multilingual NLP. Proceedings of the Seventeenth Conference on Computational Natural Language Learning, ACS. August 2013.

[2] Thamar Solorio and Yang Liu. 2008. Learning to predict code-switching points. In Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, pages 973–981. Association for Computational Linguistics.

[3] Mario Piergallini et al. 2016, Word-Level Language Identification and Predicting Codeswitching Points in Swahili-English Language Data. In Proceedings of the Second Workshop on Computational Approaches to CodeSwitching, pages 21-29. Association for Computational Linguistics.

[4] Marco Bonzanini, 2017.

[5] Scott Robinson. K-Nearest Neighbors Algorithm in Python and Scikit-Learn. 2018.

[6] Carol Myers-Scotton. 1993b. Social Motivations for Codeswitching: Evidence from Africa. Oxford University Press, Oxford, UK.

[7] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[8] Wikipedia, The Free Encyclopedia, s.v. "Topic Model," (accessed August 11, 2018), https://en.wikipedia.org/wiki/Topic_model

[9] E. Mayfield, D. Adamson, R. Enand (2014, May), Computational Linguistics, Available: http://www.lighsidelabs.com.

[10] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed Representations of Words and Phrases and their Compositionality. https://arxiv.org/pdf/1310.4546.pdf. Accepted to NIPS 2013.

[11] Shamir, G.I.. (2015). Minimum description length (MDL) regularization for online learning. Proceedings of the 1st International Workshop on Feature Extraction: Modern Questions and Challenges at NIPS 2015, in PMLR 44:260-276

[12] "JamiiForums." JamiiForums | The Home of Great Thinkers. Web.