Automatic Token and Turn Level Language Identification for Code-Switched Text Dialog: An Analysis Across Language Pairs and Corpora

Vikram Ramanarayanan
Educational Testing Service R&D
90 New Montgomery St, #1500
San Francisco, CA
vramanarayanan@ets.org

Robert Pugh
Educational Testing Service R&D
90 New Montgomery St, #1500
San Francisco, CA
rpugh@ets.org

Abstract

We examine the efficacy of various feature–learner combinations for language identification in different types of text-based code-switched interactions – human-human dialog, human-machine dialog, as well as monolog – at both the token and turn levels. In order to examine the generalization of such methods across language pairs and datasets, we analyze ten different datasets of code-switched text. We extract a variety of character- and word-based text features and pass them into multiple learners, including conditional random fields, logistic regressors, and recurrent neural networks. We further examine the efficacy of character-level embedding and GloVe features in improving performance and observe that our best-performing text system significantly outperforms the majority vote baseline across language pairs and datasets.

1 Introduction

Code-switching refers to multilingual speakers' alternating use of two or more languages or language varieties within the context of a single conversation or discourse in a manner consistent with the syntax and phonology of each variety (Milroy and Muysken, 1995; Wei, 2000; MacSwan, 2004; Myers-Scotton, 2006). Increasing globalization and the continued rise of multilingual societies around the world makes research and development of automated tools for the processing of code-switched speech a very relevant and interesting problem for the scientific community. In our case, an important additional motivating factor for studying and developing tools to elicit and process code-switched or crutched[1] language comes from the education domain, specifically language learning. Recent findings in the literature suggest that strategic use of code-switching of bilinguals' L1 and L2 in instruction serves multiple pedagogic functions across lexical, cultural, and cross-linguistic dimensions, and could enhance students' bilingual development and maximize their learning efficacy (Wheeler, 2008; Jiang et al., 2014). This seems to be a particularly effective strategy especially when instructing low proficient language learners (Ahmad and Jusoff, 2009). Therefore, the understanding of code-switched dialog and development of computational tools for automatically processing code-switched conversations would provide an important pedagogic aid for teachers and learners in classrooms, and potentially even enhance learning at scale and personalized learning.

Automated processing of code-switched text dialog poses an interesting, albeit challenging problem for the scientific community. This is because the hurdles observed during traditional dialog processing tasks such as spoken language understanding (SLU), natural language generation (NLG) and dialog management (DM) are exacerbated in the case of code-switched text where the language the speaker is using at any given instant is not known apriori. Integrating an explicit *language identification* (or LID) step into the processing pipeline can potentially alleviate these issues. Take for example a use case of designing conversational applications for non-native English language learners (ELLs) from multiple native language (or L1) backgrounds. Many such learners tend to "crutch" on their L1 while speaking in the target language (or L2) that they are learning, es-

[1] Crutching refers to language learners relying on one language to fill in gaps in vocabulary or knowledge in the other (OConnor and Crawford, 2015).

Proceedings of the SIGDIAL 2018 Conference, pages 80–88,
Melbourne, Australia, 12-14 July 2018. ©2018 Association for Computational Linguistics

pecially if they are low proficiency learners (Littlewood and Yu, 2011), resulting in mixed-language speech. In such a case, LID becomes particularly important for SLU and DM, where the dialog designer/language expert may want the conversational agent to perform different dialog actions depending on whether the speaker used his/her L1 alone, the L2 alone, or a mixture of both during the previous turn.

Researchers have made significant progress in the automated processing of code-switched text in recent years (Solorio et al., 2014; Bali et al., 2014; Molina et al., 2016). While Joshi (Joshi, 1982) had already proposed a formal computational linguistics framework to analyze and parse code-switched text in the early eighties, it was not until recently that significant strides were made in the large-scale analysis of code-switched text. These have been facilitated by burgeoning multilingual text corpora (thanks largely to the rise of social media) and corpus analysis studies (see for example Solorio et al., 2014; Bali et al., 2014; Molina et al., 2016), which have in turn facilitated advances in automated processing. Particularly relevant to our work is prior art on predicting code-switch points (Solorio and Liu, 2008) and language identification (Barman et al., 2014; King and Abney, 2013). Researchers have made much progress on LID in code-switched text (tweets, in particular) thanks to recent workshops dedicated to the topic (Solorio et al., 2014; Molina et al., 2016). One of the top-performing systems used character n-gram, prefix and suffix features, letter case and special character features and explored logistic regression and conditional random field (CRF) learners to achieve the best performance for Spanish-English codeswitched text (Shirvani et al., 2016). Yet another successful system leveraged bi-directional long short term memory networks (BLSTMs) and CRFs (along with word and character embedding features) on both Spanish-English and Standard Arabic-Egyptian language pairs (Samih et al., 2016).

While there is comparatively less work in the literature on automated analysis of code-switched speech and dialog, the number of corpora and studies is steadily growing in several language pairs – for instance, Mandarin–English (Li et al., 2012; Lyu et al., 2015), Cantonese–English (Chan et al., 2005) and Hindi–English (Dey and Fung, 2014). As far as dialog is concerned, the Bangor Corpus consists of human-human dialog conversations in Spanish–English, Welsh–English and Spanish–Welsh (Donnelly and Deuchar, 2011). More recently, Ramanarayanan and Suendermann-Oeft (2017) also proposed a multimodal dialog corpus of human-machine Hindi–English and Spanish–English code-switched data. In order to understand how turn-level LID systems for dialog perform across different languages and corpora, this paper explores the efficacy of different *text*-based features on multiple human–human and human–machine dialog corpora of code-switched data in multiple language pairs. To that end, this paper builds on other recent work that examined this phenomenon for the Bangor Miami Corpus of English–Spanish human–human dialog (Ramanarayanan et al., 2018) and expands it significantly (note however, that this study does not examine speech data). To our knowledge, this is the first such comprehensive exploration of turn-level LID performance in human-human code-switched text dialog. With that in mind, the specific contributions of this paper are to examine:

1. The performance of: (i) a range of text features (including word- and character-level embedding features) for (ii) both word-level and turn-level LID;
2. How generalizable these features are across different datasets comprising different language pairs and styles of codeswitched text – human-human dialog, human-machine dialog and monolog (tweets);
3. Turn-level LID performance by (i) using word-level LID followed by aggregation over the entire turn v.s. (ii) directly training classifiers at the turn-level.

The rest of this paper is organized as follows: Section 2 describes the various corpora used for our turn-level LID experiments. We then elucidate the various featuresets and learners we explored in Sections 3 and 4 respectively, followed by details of the experimental setup in Section 5. Next, Section 6 presents the results of our LID experiments as well as analyses of performance numbers across featureset-learner combinations, language pairs and dataset style. Finally, we conclude with a discussion of current observations and an outlook for future work in Section 7.

2 Data

We used a total of ten code-switched corpora for our experiments across language pairs and interaction type, summarized briefly below. Note that although some of these corpora contain speech as well, we only consider the text transcripts for the purposes of this paper.

- Bangor University in Wales has assembled three corpora of *human-human code-switched dialog*[2]: (i) The Miami corpus of code-switched English and Spanish, (ii) the Siarad corpus of English and Welsh, and (iii) the Patagonia corpus of Spanish and Welsh.

- The SEAME corpus[3] comprises approximately 192 hours of Mandarin-English code-switching *human-human dialog* from 156 speakers with associated transcripts (Lyu et al., 2015). The speakers were gender-balanced (49.7% female, 50.3% male) and between 19 and 33 years of age. Over 60% of the speakers were Singaporean; the rest were Malaysian.

- The HALEF corpora of code-switched *human-machine dialog* comprise English–Hindi and English–Spanish language pairs. In each language pair, bilingual human participants were encouraged to use code-switched speech as they interacted with a cloud-based multimodal dialog system to order food and drink from a virtual coffee shop barista. For more details, see Ramanarayanan and Suendermann-Oeft (2017).

- Finally, in addition to these dialog corpora, we also used monolog corpora for comparison – four Twitter datasets used in the 1st shared task on language identification held at EMNLP 2016 (Solorio et al., 2014). These consisted of code-switched tweets in the following language pairs: English–Spanish, English–Mandarin, English–Nepalese, and Modern Standard Arabic–Egyptian Arabic.

The transcripts were processed by performing whitespace tokenization on each turn, and removing event descriptions (such as "&=laugh") and

Language Pair	ENG-SPA			ENG-CHI		ENG-WEL	WEL-SPA	ENG-HIN	ENG–NEP	MSA–EGY
Corpus Name	Bangor	HALEF	Twitter	SEAME	Twitter	Bangor	Bangor	HALEF	Twitter	Twitter
Type of interaction	H2H dialog	H2M dialog	Text monolog	H2H dialog	Text monolog	H2H dialog	H2H dialog	H2M dialog	Text monolog	Text monolog
Number of turns collected	39660	582	10491	110145	910	61002	34436	727	9662	5115
Utterance-level language use or codeswitching percentage	ENG: 65% SPA: 29% CS: 6%	ENG: 36% SPA: 51% CS: 13%	ENG: 54% SPA: 18% CS: 28%	ENG: 26% CHI: 20% CS: 54%	ENG: 7% CHI: 40% CS: 53%	ENG: 3% WEL: 86% CS: 11%	WEL: 77% SPA: 18% CS: 5%	ENG: 32% HIN: 35% CS: 33%	ENG: 12% NEP: 15% CS: 73%	MSA: 74% EGY: 6% CS: 20%

Table 1: *Statistics of the different code-switching corpora considered in this paper. Note that H2H stands for human-to-human while H2M stands for human-to-machine.*

[2] http://bangortalk.org.uk/

[3] https://catalog.ldc.upenn.edu/ldc2015s04

unintelligible tokens. For the Twitter datasets, in order to enable cross-dataset comparison, we normalized the tag sets by creating an “other” class that included all tokens not belonging to either of the two relevant languages (NEs, ambiguous tokens, etc).

3 Feature Extraction

3.1 Low-Level Text Features

Following earlier work (Shirvani et al., 2016; Samih et al., 2016), we experimented with the following low-level binary text features that capture the presence or absence of the following:

- **Word n-grams**: We used a bag-of-words representation, trying uni- and bi-grams.
- **Character n-grams**: The set of unique character n-grams ($1 \leq n \leq 4$), without crossing word-boundaries. For example, the word sequence “la sal” would produce the following character n-grams {‘l’, ‘a’, ‘s’, ‘al’, ‘la’, ‘sa’, ‘sal’}.
- **Character Prefixes/Suffixes**: All affixes with length ≤ 3. For example, the word “intricate” would have prefixes {”i”, ”in”, ”int”}, and suffixes {”ate”, ”te”, and ”e”}.
- **Dictionary Lookup**: We examine whether each word exists in a dictionary for either one of the code-switched languages. Dictionaries for English, Spanish, and Welsh, were sourced from GNU Aspell[4]. Dictionaries from other languages were not used either because they were not available or the dictionary’s orthography differed from that used in our data.

We also extracted turn length (in number of words) and used that as an additional feature.

3.2 Embedding Features

We also examined the utility of different combinations of the following embedding features:

- **word2vec** based pre-trained word embeddings (Mikolov et al., 2013). These models are shallow, two-layer neural networks that represent (embed) words in a continuous vector space where semantically similar words are embedded close to each other. In order to pre-train word2vec models while analyzing the code-switched corpus of a particular language pair, we utilized other corpora (if they existed) for that same language pair (in other words, we were not able to analyze the effect of these features for language pairs that had just one exemplar corpus, like English–Welsh).
- **char2vec** based pre-trained character embeddings. These features are similar to word2vec, but are applied at the character level. In order to generate these embeddings, we run standard Word2Vec with skip-grams, except characters take the place of words and words take the place of sentences, enabling us to learn character contexts within words. Jaech et al. (2016) used a similar feature they termed “char2vec”, which is however different from our implementation; it involves learning a character-based word vector using a convolutional neural network.
- **GloVe** based pre-trained word embeddings (Pennington et al., 2014). GloVe is an unsupervised learning algorithm for obtaining vector representations for words, which capture linear substructures of interest in the word vector space. It is a global log bilinear regression model that combines the advantages of the two major methods: matrix factorization of global corpus co-occurence statistics and local context window methods. As in the *word2vec* case, to obtain pre-trained vectors for a corpus in a given language pair, we used the other corpora for that pair to train aggregated global word-word co-occurrence statistics.
- **GloVe** based pre-trained character embeddings. This is the GloVe algorithm applied at the character level. To our knowledge, our paper is the first such application of these features for language identification.
- **No pre-training**: In this case, we learned word and/or character embeddings from scratch, i.e., we randomly initialized the vectors and trained these embeddings using the training partition of the data for each cross-validation fold.

[4] http://aspell.net/

4 Machine Learning Methods

Following previous work in this area, we examined the utility of the following learners:

- **Logistic Regression**: The simplest method we investigated was just logistic regression with L2-regularization to generate language label probabilities using the various combinations of the features described in Section 3.1.
- **CRFs**: In this case, instead of modeling language tagging decisions for each word independently, we model them jointly using a conditional random field or CRF (Lafferty et al., 2001).
- **Bidirectional LSTMs**: Long short term memory networks or LSTMS are a special kind of recurrent neural network that is capable of learning long-term dependencies (Hochreiter and Schmidhuber, 1997). They do so using several gates that control the proportion of the input to give to the memory cell, and the proportion from the previous state to forget[5]. We implemented the Stack LSTM architecture first proposed by Dyer et al. (2015), in which the LSTM is augmented with a "stack pointer." While sequential LSTMs model sequences from left to right, Stack LSTMs permit embedding of a stack of objects that are both added to (using a push operation) and removed from (using a pop operation). This allows the Stack LSTM to work like a stack that maintains a "summary embedding" of its contents. In our case, we use this architecture to model a summary embedding of characters within an model of word embedding sequences. In addition to this Stack BiLSTM, following Lample et al. (2016), we used a combination of a Stack BiLSTM with a CRF, where instead of directly using the softmax output from the Stack BiLSTM, we use a CRF to predict the final language tag for each word by taking into account neighboring tags.

An novel feature of our experiments is the examination of the utility of pre-trained *GloVe* and *char2vec* in improving performance of the system proposed for named entity recognition in Lample et al. (2016).

5 Experiments

We conducted 10-fold cross-validation experiments for all datasets. For each dataset, we first extracted the word and character level features described in Section 3. We then tried the following approaches to predicting one of 3 classes – English, Spanish or Code-switched – at the turn-level: (i) Used a CRF to make word-level predictions, and aggregated them to form a turn-level prediction; (ii) aggregated the features at the turn level and try a variety of learners, including logistic regression and deep neural networks to make language predictions at the turn level; (iii) fed word- and character-embedding combinations (both with and without pre-training) to a Stacked-BiLSTM-CRF system and made an LID prediction for each turn. We experimented with different learner configurations and parameter settings and summarize the best performing featureset and learner combination in the Results section. We used a grid search method to find optimal character embedding size for each dataset (among values of 25, 50 and 100). For the Stack-BiLSTM system, given the large number of architectural parameters to optimize (number of LSTM layers and recurrent units, type of optimizer, dropout, gradient clipping/normalization, minibatch size, to name a few), we chose to use the choices recommended by Reimers and Gurevych (2017), who evaluated over 50,000 different setups and found that some parameters, like pre-trained embeddings or the last layer of the network, have a large impact on the performance, while other parameters, like the number of LSTM layers or the number of recurrent units, are of relatively minor importance. We set the word-embedding size to 100 and used 25 and 100 recurrent units in the character-level and word-level BiLSTMs, respectively, following Lample et al. (2016).

6 Observations and Analysis

Table 2 lists the best performing *turn*-level LID systems, including the feature sets and model details. In each cell of the table, the top value indicates the overall weighted average F1 score, while the bottom value (in parentheses) indicates the F1 score of the code-switched class. We decided to list the latter value since this class is easily confusable with the other two, and better F1 scores for this class might give an insight into which algorithms are better at capturing the characteristics

[5] Also see http://colah.github.io/posts/2015-08-Understanding-LSTMs

System		Weighted Average F1 Scores for Each Dataset									
Featureset	Machine Learner	ENG-SPA			ENG-CHI		ENG-WEL	WEL-SPA	ENG-HIN	ENG–NEP	MSA–EGY
		Bangor	HALEF	Twitter	SEAME	Twitter	Bangor	Bangor	HALEF	Twitter	Twitter
Word n-grams, Char n-grams, Affixes, Length & Dictionary lookup	Logistic Regression	0.9525 (0.6820)	0.9324 (0.7576)	0.8143 (0.6839)	0.9931 (0.9937)	0.5786 (0.6272)	0.9647 (0.8531)	0.9706 (0.6762)	0.8765 (0.8235)	0.8442 (0.9023)	0.7556 (0.4511)
Word n-grams, Char n-grams, Affixes, Length & Dictionary lookup	CRF aggregated to turn	**0.9696** (0.8381)	0.9584 (0.8874)	0.8912 (0.8247)	0.9977 (0.9979)	0.7393 (0.7457)	**0.9676** (0.8639)	**0.9800** (0.7982)	**0.9022** (0.8553)	**0.9367** (0.9568)	0.7280 (0.3216)
Word and Char Embeddings (both from scratch)	Stacked Bi-LSTM + CRF	0.966 (0.8345)	0.9759 (0.9560)	0.884 (0.8256)	0.999 (0.9991)	**0.742** (0.7268)	0.9606 (0.8469)	0.977 (0.7828)	0.894 (0.8536)	0.932 (0.9525)	0.747 (0.4227)
Pre-trained Word Embeddings ('word2vec' in blue, otherwise 'GloVe') and Char Embeddings (from scratch)	Stacked Bi-LSTM + CRF	0.9671 (0.8438)	0.9708 (0.9308)	0.8950 (0.8421)	0.999 (0.9987)	0.7270 (0.7104)	— —	— —	— —	— —	— —
2 Pre-trained Word and Char Embeddings ('word2vec' in blue, otherwise 'GloVe')	Stacked Bi-LSTM + CRF	**0.9692** (0.8506)	**0.976** (0.9560)	**0.8953** (0.8424)	0.999 (0.9992)	0.7332 (0.7173)	— —	— —	— —	— —	— —
Best-performing turn predictions	Stacked Bi-LSTM	0.9621 (0.7962)	0.9394 (0.8235)	0.8587 (0.7770)	— —	0.6089 (0.6821)	0.9485 (0.7869)	0.9501 (0.7277)	0.8514 (0.7889)	0.7906 (0.8710)	**0.7564** (0.5280)
Majority Baseline		0.49	0.34	0.38	0.38	0.37	0.79	0.67	0.18	0.61	0.63
Random Baseline		0.38	0.34	0.35	0.35	0.37	0.43	0.41	0.34	0.39	0.40
Best performance on 1^{st} codeswitching challenge		—	—	0.822	—	0.894	—	—	—	0.977	0.417
Best performance on 2^{nd} codeswitching challenge		—	—	0.913	—	—	—	—	—	—	0.83

Table 2: Weighted average F1 scores obtained by different featureset–learner combinations on each codeswitching dataset. Notice that datasets are organized first by language pair, and then according to type of interaction (human-human vs. human-machine vs. Twitter). Each cell of the table contains two numbers: the overall weighted F1 score on top and the F1 score of the code-switched class in parentheses at the bottom. Note that we obtained performance numbers for pre-trained word and character embeddings only for language pairs with more than one dataset, i.e., ENG-SPA and ENG–CHI. Also shown for benchmarking purposes are the best tweet-level performance numbers from the 1^{st} and 2^{nd} codeswitching challenges for some of the Twitter datasets. However, note that this is *not* a completely fair comparison, because the train-test partitions in our case are different: we used only the train data from the 1^{st} code-switching challenge in order to perform 10-fold cross-validation experiments. Also see the text for more details.

System		Weighted Average Token-Level F1 Scores for Each Dataset									
Featureset	Machine Learner	ENG-SPA			ENG-CHI		ENG-WEL	WEL-SPA	ENG-HIN	ENG–NEP	MSA–EGY
		Bangor	HALEF	Twitter	SEAME	Twitter	Bangor	Bangor	HALEF	Twitter	Twitter
Word n-grams, Char n-grams, Affixes, Length & Dictionary lookup	CRF	0.9774	0.9772	0.9111	0.9989	**0.9513**	**0.9824**	0.9774	**0.9343**	**0.9601**	**0.5804**
Word and Char Embeddings (both from scratch)	Stacked Bi-LSTM + CRF	**0.9883**	0.9721	0.9394	0.9993	0.9476	0.9820	**0.9922**	0.9290	0.9579	0.5791
Pre-trained Word Embeddings ('word2vec' in blue, otherwise 'GloVe') and Char Embeddings (from scratch)	Stacked Bi-LSTM + CRF	0.9814	0.9784	**0.9437**	0.9992	0.9429	—	—	—	—	—
Pre-trained Word and Char Embeddings ('word2vec' in blue, otherwise 'GloVe')	Stacked Bi-LSTM + CRF	0.9819	**0.9788**	0.9370	0.9993	0.9478	—	—	—	—	—
Best performance on 1^{st} codeswitching challenge		—	—	0.94	—	0.892	—	—	—	0.959	0.936
Best performance on 2^{nd} codeswitching challenge		—	—	0.973	—	—	—	—	—	—	0.876

Table 3: Weighted average F1 scores for token-level predictions after 10-fold crossvalidation. Also shown for benchmarking purposes are the best token-level performance numbers from the 1^{st} and 2^{nd} codeswitching challenges. However, note that this is *not* a fair comparison, because the train-test partitions in our case are different: we used only the train data from the 1^{st} code-switching challenge in order to perform 10-fold cross-validation experiments. Also see the text for more details.

of this class. At the outset, we observe that all text systems significantly outperform the majority vote baseline (where we assign the language labels of all turns in the test set to the majority class) and the random baseline (where the language labels of all test set turns are assigned at random) by a huge margin.

One of the primary research questions we wanted to study (see the penultimate paragraph of Section 1) was how different featureset-learner combinations performed across different language pairs. We see that no particular featureset-learner combination dominated overall performance-wise, with results varying depending on the dataset and language pair in question. Interestingly, in the case of English–Spanish, where there were 3 different datasets of code-switched text, using pre-trained word and character embeddings performed at or above par all other systems. In other words, in the presence of sufficient amounts of data for pre-training, using pre-trained embedding-based systems yields the best results. Even though the overall F1 score of all embedding-based Stack Bi-LSTM systems is similar, notice that the F1 score of the code-switched class improves when we use both pre-trained word *and* character embeddings. This suggests that pretrained character embeddings are particularly useful in capturing the characteristics of code-switched language. While GloVe-based character embeddings were more useful for the human-human (Bangor) and monolog (Twitter) datasets, word2vec was better for the HALEF dataset of human-machine dialog. For English–Mandarin corpora, on the other hand, while the embedding–Stack BiLSTM–CRF combination still performed best, using pre-trained embeddings did not seem to make any significant additional impact.

Another research question of interest dealt with whether we obtained a better turn-level LID performance by (i) using word-level LID followed by aggregation over the entire turn, or (ii) directly training classifiers at the turn-level. Our results seem to suggest that the former is better than the latter across all code-switched text datasets with one notable exception. In the case of the Modern Standard Arabic–Egyptian Arabic Twitter dataset, using a Stacked BiLSTM with embedding features and a direct softmax layer for turn-level predictions (i.e., without an additional CRF aggregation step) performed best.

For all other remaining language pairs (each of which had just one dataset), the simpler CRF classifier (where predictions were aggregated to a turn) with a more standard featureset (word and character n-grams, affixes, turn length and dictionary lookup) yielded the best results. That this simpler CRF system performed competently even in the other cases relative to the Stack-BiLSTM systems suggests that the former is perhaps a better choice when one does not have large amounts of training data, particularly for pre-training. On a related note, it is also worth pointing out that unsurprisingly, performance numbers across the board are influenced by the amount of data in each dataset, i.e., more data leads to higher F1 scores.

Yet another research question dealt with the performance across datasets for human–human dialog vs. human–machine dialog vs. monolog tweets. We observe, in general, a decrease in overall weighted F1 score as one moves from human–human dialog to human–machine dialog to monolog tweet data. One possible reason for this is that Twitter data in particular consists of many "other" non-language tokens (such as named entities, ambiguous tokens, etc.), which, on removal or non-consideration, might lead to different phrase structures in the resulting data[6].

The final question we asked was to examine token-level prediction performance, in order to benchmark ourselves against prior art in this area. Table 3 lists these results. We find that performance trends in this case roughly mirror those observed at the turn-level.

Performances from the 1st and 2nd Code-switching Workshop Challenge results are provided in each table to provide some comparison with our systems. However, it should be noted that these comparisons are not exact. Our results are from 10-fold cross-validation on the training data used in the Workshop challenge, not on the held-out test sets. Additionally, because we pulled the data from Twitter years after the 1st Workshop, some of the tweets initially intended for the dataset were no longer available. For the tweet-level performance, we report results on three-class

[6] A big part of the errors made by crowd-sourcing annotators who assigned tag labels for the Twitter datasets involve named entities, probably because the annotators do not take the context into account in an effort to be fast and collect money quickly. The problem is exacerbated in the MSA-EGY set due to the fact that there is inherently considerable amount of data overlap due to homographs between the two varieties of the language (Molina et al., 2016).

classification (language 1 vs. language 2 vs. code-switched), whereas the Code-switching Workshop performances are based on binary classification (monolingual vs. code-switched). Furthermore, as mentioned earlier, in order to enable cross-dataset comparison, we normalized the tag sets by creating an ”other” class that included all tokens not belonging to either of the two relevant languages (NEs, ambiguous tokens, etc). Taking these points into consideration, our systems perform competitively with the submissions to the Code-switching Workshop Challenges. The only exception is in the case of the MSA-EGY dataset, where while our tweet-level performance is competitive, our token-level performance far underperforms the state-of-the-art. We suspect that dataset imbalance could play a role, as well as the fact that we didn’t use any external resources for this language pair.

7 Discussion and Outlook

We have presented an experimental evaluation of different text-based featuresets in performing language identification (LID) at both the turn and token levels in code-switched text interactions. We studied the generalizability of various systems both across language pair and dataset type—human–human, human–machine and monolog—by examining 10 different datasets of code-switched text. While our best text-based systems performed either at or above par with the state of the art in the field, we found that the use of both pre-trained word and character-based embedding features, and the latter in particular (either through *char2vec* or *GloVe*), were particularly useful at capturing the characteristics of code-switched speech (with the caveat that the feature extraction process requires sufficient data for pre-training). We further observed that a performance drop depending on the style of interaction, as we move from human–human dialog to human–machine dialog to monolog tweets.

Going forward, we will explore a number of potential avenues for improving the performance of the text-based LID systems. Chief among these is to investigate strategies for dealing with little or no code-switched data (or indeed, overall training data) for a given language pair, and how to improve the performance of deep learning algorithms for such datasets. In addition, we would like to perform a deeper error analysis of the algorithms on different featuresets to obtain a better understanding of how best to select a feature-learner combination for the LID task.

Finally, as mentioned earlier, one of the key exciting R&D directions that this work informs is in building code-switching dialog systems. For instance, integrating an explicit language identification step into the spoken language understanding (SLU) could help enhance the system performance. Over and above such applications, such an LID module might also help inform pragmatic considerations during dialog management and the language generation module for the generation of appropriate mixed-language output.

8 Acknowledgements

We would like to thank Julia Hirschberg and other attendees at the Interspeech 2018 Special Session on Speech Techologies for Code-switching in Multilingual Communities for useful discussions and illuminating ideas on the processing of code-switched speech, including available datasets.

References

Badrul Hisham Ahmad and Kamaruzaman Jusoff. 2009. Teachers code-switching in classroom instructions for low english proficient learners. *English Language Teaching* 2(2):49.

Kalika Bali, Yogarshi Vyas, Jatin Sharma, and Monojit Choudhury. 2014. i am borrowing ya mixing? an analysis of english-hindi code mixing in facebook. *Proceedings of the First Workshop on Computational Approaches to Code Switching, EMNLP 2014* page 116.

Utsab Barman, Amitava Das, Joachim Wagner, and Jennifer Foster. 2014. Code mixing: A challenge for language identification in the language of social media. *EMNLP 2014* 13.

Joyce YC Chan, PC Ching, and Tan Lee. 2005. Development of a cantonese-english code-mixing speech corpus. In *INTERSPEECH*. pages 1533–1536.

Anik Dey and Pascale Fung. 2014. A hindi-english code-switching corpus. In *LREC*. pages 2410–2413.

Kevin Donnelly and Margaret Deuchar. 2011. The bangor autoglosser: a multilingual tagger for conversational text. *ITA11, Wrexham, Wales* .

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A Smith. 2015. Transition-based dependency parsing with stack long short-term memory. *arXiv preprint arXiv:1505.08075* .

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Aaron Jaech, George Mulcaire, Mari Ostendorf, and Noah A Smith. 2016. A neural model for language identification in code-switched tweets. In *Proceedings of The Second Workshop on Computational Approaches to Code Switching*. pages 60–64.

Yih-Lin Belinda Jiang, Georgia Earnest García, and Arlette Ingram Willis. 2014. Code-mixing as a bilingual instructional strategy. *Bilingual Research Journal* 37(3):311–326.

Aravind K Joshi. 1982. Processing of sentences with intra-sentential code-switching. In *Proceedings of the 9th conference on Computational linguistics-Volume 1*. Academia Praha, pages 145–150.

Ben King and Steven P Abney. 2013. Labeling the languages of words in mixed-language documents using weakly supervised methods. In *HLT-NAACL*. pages 1110–1119.

John D Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*. Morgan Kaufmann Publishers Inc., pages 282–289.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT*. pages 260–270.

Ying Li, Yue Yu, and Pascale Fung. 2012. A mandarin-english code-switching corpus. In *LREC*. pages 2515–2519.

William Littlewood and Baohua Yu. 2011. First language and target language in the foreign language classroom. *Language Teaching* 44(1):64–77.

Dau-Cheng Lyu, Tien-Ping Tan, Eng-Siong Chng, and Haizhou Li. 2015. Mandarin–english code-switching speech corpus in south-east asia: Seame. *Language Resources and Evaluation* 49(3):581–600.

Jeff MacSwan. 2004. Code switching and grammatical theory. *The handbook of bilingualism* 46:283.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* .

Lesley Milroy and Pieter Muysken. 1995. *One speaker, two languages: Cross-disciplinary perspectives on code-switching*. Cambridge University Press.

Giovanni Molina, Nicolas Rey-Villamizar, Thamar Solorio, Fahad AlGhamdi, Mahmoud Ghoneim, Abdelati Hawwari, and Mona Diab. 2016. Overview for the second shared task on language identification in code-switched data. *EMNLP 2016* page 40.

Carol Myers-Scotton. 2006. Codeswitching with english: types of switching, types of communities. *World Englishes: Critical Concepts in Linguistics* 4(3):214.

Brendan H OConnor and Layne J Crawford. 2015. An art of being in between: The promise of hybrid language practices. In *Research on Preparing Inservice Teachers to Work Effectively with Emergent Bilinguals*, Emerald Group Publishing Limited, pages 149–173.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. pages 1532–1543.

Vikram Ramanarayanan, Robert Pugh, Yao Qian, and David Suendermann-Oeft. 2018. Automatic Turn-Level Language Identification for Code-Switched Spanish-English Dialog. In *Proc. of the IWSDS Workshop 2018, Singapore*.

Vikram Ramanarayanan and David Suendermann-Oeft. 2017. Jee haan, I'd like both, por favor: Elicitation of a Code-Switched Corpus of Hindi–English and Spanish–English Human–Machine Dialog. *Proc. Interspeech 2017* pages 47–51.

Nils Reimers and Iryna Gurevych. 2017. Optimal hyperparameters for deep lstm-networks for sequence labeling tasks. *arXiv preprint arXiv:1707.06799* .

Younes Samih, Suraj Maharjan, Mohammed Attia, Laura Kallmeyer, and Thamar Solorio. 2016. Multilingual code-switching identification via lstm recurrent neural networks. *EMNLP 2016* page 50.

Rouzbeh Shirvani, Mario Piergallini, Gauri Shankar Gautam, and Mohamed Chouikha. 2016. The howard university system submission for the shared task in language identification in spanish-english codeswitching. In *Proceedings of The Second Workshop on Computational Approaches to Code Switching*. pages 116–120.

Thamar Solorio, Elizabeth Blair, Suraj Maharjan, Steven Bethard, Mona Diab, Mahmoud Gohneim, Abdelati Hawwari, Fahad AlGhamdi, Julia Hirschberg, Alison Chang, et al. 2014. Overview for the first shared task on language identification in code-switched data. In *Proceedings of the First Workshop on Computational Approaches to Code Switching*. Citeseer, pages 62–72.

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

Li Wei. 2000. *The bilingualism reader*. Psychology Press.

Rebecca S Wheeler. 2008. Code-switching. *EDUCATIONAL LEADERSHIP* .