Comparing Grammatical Theories of Code-Mixing

Adithya Pratapa[†]
Carnegie Mellon University
vpratapa@cs.cmu.edu

Monojit Choudhury
Microsoft Research
monojitc@microsoft.com

Abstract

Code-mixed text generation systems have found applications in many downstream tasks, including speech recognition, translation and dialogue. A paradigm of these generation systems relies on well-defined grammatical theories of code-mixing, and there is a lack of comparison of these theories. We present a large-scale human evaluation of two popular grammatical theories, Matrix-Embedded Language (ML) and Equivalence Constraint (EC). We compare them against three heuristic-based models and quantitatively demonstrate the effectiveness of the two grammatical theories.

1 Introduction

Code-mixing is the phenomenon of mixing multiple languages within a single conversation. While widely observed in spoken language, mixing languages has also become commonplace in informal text conversations with the increasing use of social media (Rijhwani et al., 2017). Prior work has focused on various code-mixed natural language processing (NLP) tasks, including part-of-speech (POS) tagging (Soto and Hirschberg, 2017; Singh et al., 2018), sentiment analysis (Patwa et al., 2020), machine translation (Solorio et al., 2021) and speech recognition (Lee et al., 2017). Accurately modeling when and how to mix languages is critical to the above code-mixed NLP tasks.

Research efforts on this front have either relied on established grammatical theories (Li and Fung, 2012; Bhat et al., 2016) or neural sequence-to-sequence models (Winata et al., 2019) to generate realistic code-mixed text. Such models have found applications in speech recognition (Li and Fung, 2012; Lee et al., 2019), translation (Gupta et al., 2021) and other downstream NLP tasks (Pratapa et al., 2018b). These generation techniques have also been applied in human-machine dialogue (Ahn et al., 2020; Bawa et al., 2020) to generate seemingly natural code-mixed responses.

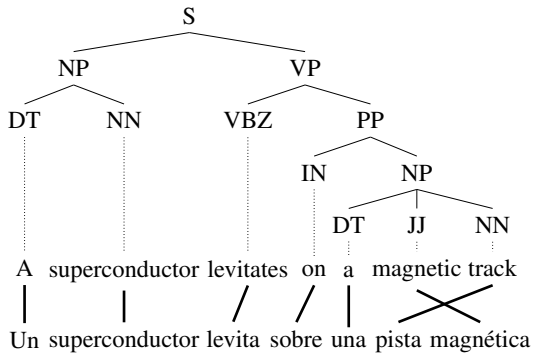
However, current literature lacks a thorough comparison of different code-mixed text generation models. Such comparison is necessary to understand the relevance of these generation models to individual downstream NLP applications. In this work, we present one such comparison involving two linguistic theory-based code-mixed text generation systems, one based on Equivalence Constraint theory (Poplack, 1980) and the other on Matrix-Embedded language theory (Myers-Scotton, 1993). Specifically, we crowdsource human judgments on the understandability and naturalness of text generated by the two systems. Our results show that grammatical models are considerably better than heuristic counterparts. Additionally, we find the two grammatical models are equally preferable by our human judges. We evaluate the importance of individual constraints in each grammatical theory. In the following sections, we first briefly describe the grammatical models (§2.1) and related heuristic models (§2.2). In §3, we describe our human evaluation setup, and present the results in §4.

2 Generative Models of Code-Mixing

Code-mixed text generation has applications in NLP tasks of speech recognition, human-machine dialogue, and translation. Prior work has primarily focused on three grammatical theories, Equivalence Constraint (EC) (Poplack, 1980), Matrix-Embedded Language (ML) (Myers-Scotton, 1993) and Functional Head Constraint (FHC) (Belazi et al., 1994). Each of these theories presents an account of the grammatical constraints of code-mixing. Bhat et al. (2016) presented computational models for EC and ML theories. Such computational models allow for generation of grammatically correct code-mixed text (Li and Fung, 2012; Pratapa et al., 2018a; Lee et al., 2019; Tarunesh et al., 2021). However, the resulting text may or

† work done at Microsoft Research.

Proceedings of the 2021 EMNLP Workshop W-NUT: The Seventh Workshop on Noisy User-generated Text, pages 158–167



(a) English constituency parse + word-level alignments

1	Un superconductor levita sobre *a magnetic track*
2	Un superconductor levita sobre una pista *magnetic*
3	Un superconductor levita sobre *a* pista magnética
4	A superconductor levitates on a magnetic *magnética*
5	Levita un superconductor sobre *a magnetic track*

(b) Candidate code-mixed sentences.

Figure 1: An illustration of outputs from code-mixed text generation models. Figure 1a shows the constituency parse for an example English sentence, along with its word-aligned Spanish translation. Figure 1b presents 5 example code-mixed sentences generated from the above English-Spanish aligned pair. Sentence #1 is allowed under all the grammatical and heuristic models, #2 is disallowed by EC model, #3 is disallowed by ML model, #4 is disallowed by EC, ML, aligned and dictionary models, #5 is disallowed under all the grammatical and heuristic models. Words from embedded language are *italicized* in each example.

may not be considered natural by bilingual speakers. In this work, we compare the two theories (EC and ML) for their perceived naturalness. Due to the unavailability of a computational model for FHC theory, we leave the comparison of FHC to EC and ML theories to future work.

There is another line of work involving sequence-to-sequence models that do not directly rely on grammatical theories (Winata et al., 2019; Lee and Li, 2020). While this is an exciting line of research, we restrict the focus of this work to grammatical models and leave the comparison with neural models to future work.

2.1 Grammatical Models

Equivalence Constraint (EC): Proposed by Poplack (1980), this theory imposes an *equivalence* constraint at each switch point between the two languages in a code-mixed sentence. The two constituent languages are assumed to follow context-free grammar (CFG). Each non-terminal (and terminal) in one CFG has a counterpart in the other CFG. In cases where the two language parses are not identical, we follow prior work (Bhat et al., 2016; Pratapa et al., 2018a) and perform node collapse to make the grammars equivalent. We follow the implementation from Pratapa et al. (2018a) to generate code-mixed sentences for our study.[1]

Matrix-Embedded Language (ML): Proposed by Myers-Scotton (1993), this theory allows for inserting grammatical constituents of an *embedded* language into a sentence in *matrix* language. It disallows for replacing specific constituents relating to verbs (V), coordinating conjunction (CC), determiner (DT), preposition (IN) as well as nesting sub-trees (NST).

We refer the readers to Bhat et al. (2016) for more details on the computational models for EC and ML theories.

2.2 Heuristic Models

The above described grammatical models impose linguistic constraints on the process of mixing text from two languages. To evaluate if these linguistic constraints are necessary for the output text to be considered natural, we contrast them against three simple heuristic-based models. These heuristic models only rely on parallel sentences, word-level alignments, or a bilingual dictionary.

Aligned ($\mathbf{H_{aligned}}$): Given parallel sentences with word-level alignments, we construct a code-mixed sentence by uniformly sampling a word from one of the two languages in a given aligned pair. To generate a sentence, we first sort the alignment indices in the order of L1 tokens (similarly for L2) and then iterate through the alignment indices.

Dictionary ($\mathbf{H_{dict}}$): In this model, we use an external bilingual dictionary that maps words from L1 and L2. We take a monolingual sentence, and for every word, we uniformly sample one of the words or its counterpart from the other language (if one exists in the dictionary). This model only needs a

[1] See Rizvi et al. (2021) for the implementation.

pre-built dictionary to synthesize code-mixed sentences from monolingual corpora.

Parallel ($H_{parallel}$): In the case of missing word-level alignments, we can still generate code-mixed sentences by assuming a linear mapping between L1 and L2 word order. This assumption might work well for languages with similar word order, like the English-Spanish pair used in this study.

Figure 1 illustrates the validity of example Spanish-English code-mixed sentences under each grammatical and heuristic model discussed above.

3 Evaluation Setup

To evaluate the effectiveness of the above described models of code-mixing (§2), we first sample Spanglish (English-Spanish) sentences using selected criterion (§3.1), and then collect human judgments on their understandability and naturalness (§3.2).

3.1 Preparing Data

The generative models of code-mixing take parallel sentences and their word-level alignments as their inputs. We follow the same strategy as Pratapa et al. (2018a) to collect monolingual tweets in English and Spanish.[2] We then translate the tweets to the other language using Microsoft Translator API. We use `fast_align` toolkit (Dyer et al., 2013) to extract word-level alignments between the source and target sentences.[3] For the grammatical models, we extract constituent parses using Stanford PCFG parser (Klein and Manning, 2003). For generating code-mixed sentences using the grammatical theories, we follow the prior work (Bhat et al., 2016; Pratapa et al., 2018a). Note that both these systems follow the grammatical theories exactly and do not exclude code-mixed sentences based on their naturalness. For the heuristic models, we follow the methodologies described in §2.2. Most models take parallel sentences as input and generate a large number of code-mixed sentences. For this study, we sample code-mixed sentences by carefully controlling for specific features. As we describe below, we identify features relevant to the parallel sentences as well as grammatical constraints.

Sentence length (SL): We compute the average length (# tokens) for each pair of parallel sentences. We then group them into the four buckets, {SL_1: (5, 8], SL_2: (8, 11], SL_3: (11, 15], SL_4: (15, 20]}.

Pseudo fuzzy match score (PFMS): As described earlier, we rely on a neural machine translation system to automatically translate monolingual tweets. The translation performance itself will have an impact on the perceived quality of generated code-mixed sentences. To evaluate the translation performance, we compute the normalized edit distance between the source sentence and its back-translated counterpart. We term this score as the pseudo fuzzy match score (PFMS) (Equation 1; Pratapa et al. (2018a)). PFMS lies in [0,1] and we split the data into four classes based on PFMS value, {$PFMS_1$: (0, 0.3], $PFMS_2$: (0.3, 0.5], $PFMS_3$: (0.5, 0.7], $PFMS_4$: (0.7, 1.0]}. Since tweets themselves are noisy user-generated texts, their quality will have an impact on the generated code-mixed texts.

$$\text{PFMS} = \frac{\mathit{EditDistance}(s, s')}{max(|s|, |s'|)} \tag{1}$$

Alignment complexity score (ACS): Another factor that impacts the perceived quality of generated code-mixed text is the complexity of token-level alignments between the parallel sentences. To this end, we compute ACS, that captures two important aspects, 1. *order of alignments*: mean distance between aligned pair of tokens (ACS_1), and 2. *degree of alignment*: average degree of a token (ACS_2). We have grouped the input pair into the two buckets, $(max(0, \mu - \sigma), \mu]$ and $[\mu, \mu + \sigma)$, where μ and σ denote the mean and standard deviation of the ACS scores. This resulted in the intervals [0, 0.39], (0.39, 0.98] and (0.90, 1.06], (1.06, 1.21] for ACS_1 and ACS_2 respectively. In Equation 2, i and j denote the i-th and j-th tokens in the source (of length l) and target (of length l') sentences respectively. $A(i)$ denotes the set of target tokens aligned to the i-th source token. In Equation 3, d_i denotes the degree of i-th token in the sentence.

$$\text{ACS}_1 = \frac{\sum_{i=1}^{l} |i - \frac{\sum_{j \in A(i)} j}{|A(i)|} \cdot \frac{l}{l'}|}{2l} + \frac{\sum_{j=1}^{l'} |j - \frac{\sum_{i \in A^{-1}(j)} i}{|A^{-1}(j)|} \cdot \frac{l'}{l}|}{2l'} \tag{2}$$

$$\text{ACS}_2 = \frac{\sum_{l} d_i}{2l} + \frac{\sum_{l'} d_j}{2l'} \tag{3}$$

[2] We use tweets in our study for two reasons, 1. In textual format, code-mixing is most common in informal conversations (e.g., social media posts), and 2. it's easier to collect tweets over speech transcriptions.

[3] https://github.com/clab/fast_align

Understandability	
1	No, this tweet doesn't make sense
2	Not sure, but I can guess the meaning of this tweet
3	Certainly, I get the meaning of this tweet
Naturalness	
0	I am not sure
1	unnatural, and I can't imagine people using this style of Spanglish
2	weird, but who knows, it could be some style of Spanglish
3	quite natural, but I think this style of Spanglish is rare
4	natural, and I think this style of Spanglish is used in real life
5	perfectly natural, and I think this style of Spanglish is very frequently used

Table 1: Dimensions of evaluation.

For each pair of sampled parallel sentences, we select code-mixed sentences from ML, EC theories as well as their relaxed variants.

Matrix-Embedding Language (ML): We sample code-mixed sentences generated by the ML theory. Additionally, to evaluate the impact of the individual constraints of this theory, we relax by allowing mixing of closed-class items (CC, DT, IN, V) and potential nesting (allowance to switching back to matrix language in the same sub-tree).

Equivalence constraint (EC): Similar to ML theory, we sample code-mixed sentences generated by the EC theory. We also sample from a relaxed variant where we allow for lexical substitution ($EC_{[LS]}$).

Switch point variation (SP): Prior work has highlighted the importance of controlling for # switch points (SP) in the automatically generated sentences.[4] To this end, we sample EC-theory-based code-mixed sentences with a varying number of switch points. We have grouped SPs into the buckets, {SP_1: {1}, SP_2: {2}, SP_3: {3,4}, SP_4: {5,6}, SP_5: {7... 20}}.

3.2 Human Evaluation

We ask human judges to score each code-mixed sentence on their understandability and naturalness. We conducted this study on Amazon Mechanical Turk,[5] and the goal of the study is to understand the human perception of various linguistic theories of code-mixing. Both ML and EC theories establish grammatical rules for code-mixing; therefore, this study helps determine if these rules are necessary for a code-mixed sentence to be considered intelligible and realistic.

Task Setup: Every Human Intelligence Task (HIT) consists of 10 sentences, of which 8 are synthetic Spanglish sentences while 2 are *validation* sentences. Turkers rate the sentences on their understandability and naturalness. Table 1 summarizes the rubrics for understandability (scale of 1-3) and naturalness (scale of 0-5). To ensure that the judges comprehend and score the sentences appropriately, we incorporated two control sentences, a real Spanglish tweet, and a non-Spanglish tweet. The non-Spanglish tweets are code-mixed sentences taken from other language pairs, French-English, Dutch-English, and German-English. If a sentence is incomprehensible to a Spanish-English bilingual speaker, we ask the human judge to mark it with low understandability. As a sanity check, for the first 20 HITs, we ask the judges to provide English translations to the given code-mixed sentences. The expected time to complete a HIT is less than 120 seconds based on our pilot study,[6] and we paid $0.25 per assignment per HIT. We collected three valid assignments per HIT.

Qualification requirements: To ensure that the judges are Spanish-English bilingual speakers, we designed a custom qualification test. Our test evaluates their proficiency in English, Spanish and their ability to comprehend Spanglish text. The qualification test consisted of two tasks, 1. word-level language identification and 2. naturalness assessment.

[4] A switch-point denotes a word boundary in a given sentence where the languages of the words on two sides differ.

[5] https://www.mturk.com/

[6] We observed similar response times in the actual HITs

In the language identification task, we ask the participants to identify the language of a highlighted word in the context. In most cases, this word is part of both the English and Spanish vocabulary; so, the participant had to disambiguate it using the contextual information. We also include a couple of sentences where the highlighted word is neither English nor Spanish. In the naturalness task, we ask the participants to gauge the naturalness of 5 code-mixed sentences. For this task, we collaborated with Spanish-English bilingual experts to curate a mix of perfectly natural and unnatural Spanglish sentences. In addition to passing the qualification test, we also added the standard requirements on HIT acceptance rate ($\geq$97) and # HITs ($\geq$200).[7]

4 MTurk Results & Discussion

We published a total of 429 HITs and obtained judgments from 59 unique crowd workers. In our analysis, we only consider the responses where the understandability score is strictly greater than one.[8] Before analyzing the human judgments, we ensure that the human judge has marked the appropriate responses for the control data involving real-Spanglish and non-Spanglish sentences. The non-Spanglish sentences are to be scored low on the understandability scale. Naturalness being a subjective marker, there is no definite answer; therefore, we haven't pruned any of the HITs based on the judgments on real-Spanglish sentences.

Our main aim in this experiment was to gauge the importance of various sentence-level features in the code-mixed sentence generation process. To this end, we have compared the absolute human judgment scores across sentence pairs. We have designed our task as an absolute judgment task instead of pair-wise comparison for logistical reasons, as it allows for a large scale evaluation. We construct a matrix (M) with sentence/tweet pairs (T_i, T_j) along the rows and annotators (A_k) along the columns (Equation 4). $A_{i,k}$ denotes the score of annotator A_k for the sentence T_i.

$$M_{(i,j),k} = sgn(A_{i,k} - A_{j,k}) \tag{4}$$

We construct a *partial order* between T_i and T_j using the matrix M as follows,

$$\begin{aligned} \text{if } \forall k,\ M_{(i,j),k} = 0, &\quad T_i \simeq T_j \\ \text{else if } \forall k,\ M_{(i,j),k} > 0, &\quad T_i \succ T_j \\ \text{else if } \forall k,\ M_{(i,j),k} < 0, &\quad T_i \prec T_j \\ \text{else}, &\quad T_i \ ?\ T_j \end{aligned} \tag{5}$$

Here, $T_i \succ T_j$ means that sentence T_i is preferred to sentence T_j by all the common annotators (or coder). In this setting, we only compare sentences that have at least one common coder. We also consider an alternative approach, where we directly compare every sentence pair based on their average judgment scores. Unlike previous ordering, this comparison will result in a *total order*. To normalize for coders' variance, we use z-values in place of absolute scores (Equation 6). This methodology allows for analysis over a much larger space of sentence pairs.

$$\begin{aligned} z_{i,k} &= \frac{x_{i,k} - \mu_k}{\sigma_k} \\ T_i > T_j, &\quad \text{if } \sum_k z_{i,k} > \sum_k z_{j,k} \end{aligned} \tag{6}$$

To measure the inter-coder agreement, we have used Krippendorff's α as it allows for missing data

M1	M2	#(?)	#($\succ$)	#($\simeq$)	#($\prec$)
ML	$ML_{[CC]}$	1	16	11	14
ML	$ML_{[DT]}$	2	35	31	28
ML	$ML_{[IN]}$	5	40	30	39
ML	$ML_{[V]}$	11	75	41	48
ML	$ML_{[NST]}$	9	93	56	79
EC	$EC_{[LS]}$	8	48	26	51
ML	EC	27	217	164	207

Table 2: Comparison of linguistic theory based models using common coders (Equation 5).

M1	M2	#($>$)	#($<$)
ML	$ML_{[CC]}$	68	54
ML	$ML_{[DT]}$	162	133
ML	$ML_{[IN]}$	199	155
ML	$ML_{[V]}$	304	212
ML	$ML_{[NST]}$	331	303
EC	$EC_{[LS]}$	203	179
ML	EC	962	930

Table 3: Comparison of linguistic theory based models on all tweet pairs (Equation 6).

[7] see A.1 in Appendix for the questions used in the test.

[8] We acknowledge that understandability is a requirement for a code-mixed text generation model. But in the scope of this work, we focus primarily on the naturalness evaluation.

and multiple coders (Artstein and Poesio, 2008). With $d(x, y) = max(0, -sgn(x * y))$ as the distance metric, i.e., penalizing only if x and y are of different signs, we observed the α to be 0.59, indicating moderate to substantial agreement (Landis and Koch, 1977). We use the partial order in the sentences to learn a partial order for the feature variants. When comparing the impact of two feature variants, we kept the other feature values constant for consistency. Table 2 and Table 3 present a comparison of original theories (ML and EC) with their relaxed variants. The results show that the original ML theory is better than its relaxation(s). However, we observe no such preference in the EC theory. Notably, there is no clear winner between the original ML and EC theories.

Additionally, we analyze the impact of sentence length (SL), translation quality (PFMS), and # switch points (SP) on the naturalness. Our observations are summarized below,

- $SL_1 \succ SL_2 \succ SL_3 \succ SL_4$
- $PFMS_1 \prec PFMS_2 \prec PFMS_3 \prec PFMS_4$
- $SP_1 \succ SP_2 \succ SP_3 \succ SP_4$

Notably, human judges prefer fewer switch points in the code-mixed texts. This result is in line with observations from Pratapa et al. (2018a). They found the fraction of switch points in real code-mixed tweets to be ∼0.1, indicating speakers typically use few switch points. As expected, speakers prefer sentences with higher PFMS scores (i.e., translation quality). Additionally, we find that speakers prefer shorter sentences, possibly because longer sentences tend to be noisier.

To better understand the overall impact of the linguistic-theory-based constraints, we also ran ∼30 HITs consisting of synthetic data generated using heuristic-based models. For every feature (SL, ACS, PFMS), we sample one-sentence pair. We only considered PFMS ∈ $PFMS_4$ for best comparison. Since SP value was a crucial factor in determining the quality of the synthetic code-mixed sentence, we sampled sentences (heuristic-based) from the same SP brackets as the previous HIT data (linguistic-theory based). Note that, in the MTurk analysis, we have only considered those sentences acceptable under heuristic models but are invalid according to all the linguistic theories (original and relaxed) for meaningful comparison. We compared across all tweet pairs (Equation 6) as the number of common coders is low. While comparing models $H_{aligned}$, H_{dict} and $H_{parallel}$ with the linguistic models, we controlled for SP and other monolingual sentence-level features.

M1	M2	#(>)	#(<)
ML	$H_{aligned}$	109	60
$ML_{[V]}$	$H_{aligned}$	60	47
EC	$H_{aligned}$	162	75
$EC_{[LS]}$	$H_{aligned}$	31	31
ML	H_{dict}	151	58
$ML_{[V]}$	H_{dict}	70	49
EC	H_{dict}	215	81
$EC_{[LS]}$	H_{dict}	43	22
ML	$H_{parallel}$	126	58
$ML_{[V]}$	$H_{parallel}$	65	43
EC	$H_{parallel}$	188	86
$EC_{[LS]}$	$H_{parallel}$	42	19

Table 4: Comparison of heuristic based models $H_{aligned}$, H_{dict} and $H_{parallel}$ (exclusive) with linguistic models in the MTurk experiment.

In general, we found the original linguistic models to be much better than heuristic models (Table 4). While the original EC theory is much better than the heuristic model $H_{aligned}$, the relaxed version of $EC_{[LS]}$ is as good as the heuristic model. However, ML theory sentences, even after the relaxation of allowing for verb phrase switching, are better than $H_{aligned}$. We find all the linguistic theory-based models to be much better than heuristic models H_{dict} and $H_{parallel}$.

In Table 5, we compute the overlap in sentences generated by different models. As expected, both grammatical models allow fewer code-mixed sentences than heuristic models, with ML being the most constrained model. EC theory overlaps significantly with $H_{aligned}$ (54%). The two grammatical theories only overlap in 22% of the sentences.

5 Conclusion & Future Work

We present a large-scale qualitative comparison of different code-mixed generation systems. Through the judgments of Spanish-English bilingual speakers, we analyze the need for each constraint in the two grammatical theories, Matrix-Embedded, and Equivalence Constraint. We also contrast against simpler heuristic systems and found grammatical systems to be superior. However, we do not observe a clear winner between the two grammatical

M1	**M2**	$\mathbf{S_{M1} - S_{M2}}$	$\mathbf{S_{M1} \cap S_{M2}}$	$\mathbf{S_{M2} - S_{M1}}$
ML	EC	0.07	0.22	0.71
ML	ML $\cup$ ML$_{[CC]}$	0.00	0.88	0.12
ML	ML $\cup$ ML$_{[DT]}$	0.00	0.84	0.16
ML	ML $\cup$ ML$_{[IN]}$	0.00	0.75	0.25
ML	ML $\cup$ ML$_{[V]}$	0.00	0.61	0.39
EC	EC $\cup$ EC$_{[LS]}$	0.00	0.86	0.14
ML	H$_{aligned}$	0.02	0.16	0.82
EC	H$_{aligned}$	0.05	0.54	0.42
ML $\cap$ EC	H$_{aligned}$	0.00	0.16	0.84
ML $\cup$ EC	H$_{aligned}$	0.06	0.54	0.40
ML $\cup$ ML$_{[V]}$	H$_{aligned}$	0.03	0.25	0.72
EC $\cup$ EC$_{[LS]}$	H$_{aligned}$	0.06	0.57	0.38
ML	H$_{dict}$	0.09	0.03	0.88
EC	H$_{dict}$	0.25	0.06	0.69
ML $\cap$ EC	H$_{dict}$	0.07	0.03	0.90
ML $\cup$ EC	H$_{dict}$	0.26	0.06	0.68
ML $\cup$ ML$_{[V]}$	H$_{dict}$	0.15	0.04	0.82
EC $\cup$ EC$_{[LS]}$	H$_{dict}$	0.28	0.06	0.66
ML	H$_{parallel}$	0.06	0.06	0.87
EC	H$_{parallel}$	0.17	0.16	0.67
ML $\cap$ EC	H$_{parallel}$	0.05	0.06	0.89
ML $\cup$ EC	H$_{parallel}$	0.18	0.16	0.66
ML $\cup$ ML$_{[V]}$	H$_{parallel}$	0.10	0.08	0.81
EC $\cup$ EC$_{[LS]}$	H$_{parallel}$	0.20	0.16	0.64

Table 5: Comparison of sentence counts of original linguistic theories and their relaxed variants. S_{M1} and S_{M2} denote the set of sentences from models M1 and M2.

theories. Potential directions for future work include 1. extending the naturalness evaluation to sequence-to-sequence models of code-mixing, 2. expanding the analysis to other language pairs.

Acknowledgements

We thank the anonymous reviewers for their valuable feedback. We also thank the Mechanical Turk workers for their efforts in our evaluation.

References

Emily Ahn, Cecilia Jimenez, Yulia Tsvetkov, and Alan W Black. 2020. What code-switching strategies are effective in dialog systems? In *Proceedings of the Society for Computation in Linguistics 2020*, pages 254–264, New York, New York. Association for Computational Linguistics.

Ron Artstein and Massimo Poesio. 2008. Survey article: Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.

Anshul Bawa, Pranav Khadpe, Pratik Joshi, Kalika Bali, and Monojit Choudhury. 2020. Do multilingual users prefer chat-bots that code-mix? let's nudge and find out! *Proceedings of the ACM on Human-Computer Interaction*, 4(CSCW1):1–23.

Hedi M. Belazi, Edward J. Rubin, and Almeida Jacqueline Toribio. 1994. Code switching and x-bar theory: The functional head constraint. *Linguistic Inquiry*, 25(2):221–237.

Gayatri Bhat, Monojit Choudhury, and Kalika Bali. 2016. Grammatical constraints on intra-sentential code-switching: From theories to working models. *arXiv preprint arXiv:1612.04538*.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Abhirut Gupta, Aditya Vavre, and Sunita Sarawagi. 2021. Training data augmentation for code-mixed translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5760–5766, Online. Association for Computational Linguistics.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan. Association for Computational Linguistics.

J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174.

Grandee Lee, Thi-Nga Ho, Eng-Siong Chng, and Haizhou Li. 2017. A review of the mandarin-english code-switching corpus: Seame. In *2017 International Conference on Asian Language Processing (IALP)*, pages 210–213.

Grandee Lee and Haizhou Li. 2020. Modeling code-switch languages using bilingual parallel corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 860–870, Online. Association for Computational Linguistics.

Grandee Lee, Xianghu Yue, and Haizhou Li. 2019. Linguistically Motivated Parallel Data Augmentation for Code-Switch Language Modeling. In *Proc. Interspeech 2019*, pages 3730–3734.

Ying Li and Pascale Fung. 2012. Code-switch language model with inversion constraints for mixed language speech recognition. In *Proceedings of COLING 2012*, pages 1671–1680, Mumbai, India. The COLING 2012 Organizing Committee.

Carol Myers-Scotton. 1993. *Duelling Languages:Grammatical structure in Code-switching*. Clarendon Press, Oxford.

Parth Patwa, Gustavo Aguilar, Sudipta Kar, Suraj Pandey, Srinivas PYKL, Björn Gambäck, Tanmoy Chakraborty, Thamar Solorio, and Amitava Das. 2020. SemEval-2020 task 9: Overview of sentiment analysis of code-mixed tweets. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 774–790, Barcelona (online). International Committee for Computational Linguistics.

Shana Poplack. 1980. Sometimes I'll start a sentence in Spanish y termino en español. *Linguistics*, 18:581–618.

Adithya Pratapa, Gayatri Bhat, Monojit Choudhury, Sunayana Sitaram, Sandipan Dandapat, and Kalika Bali. 2018a. Language modeling for code-mixing: The role of linguistic theory based synthetic data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1543–1553, Melbourne, Australia. Association for Computational Linguistics.

Adithya Pratapa, Monojit Choudhury, and Sunayana Sitaram. 2018b. Word embeddings for code-mixed language processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3067–3072, Brussels, Belgium. Association for Computational Linguistics.

Shruti Rijhwani, Royal Sequiera, Monojit Choudhury, Kalika Bali, and Chandra Shekhar Maddila. 2017. Estimating code-switching on Twitter with a novel generalized word-level language detection

technique. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1971–1982, Vancouver, Canada. Association for Computational Linguistics.

Mohd Sanad Zaki Rizvi, Anirudh Srinivasan, Tanuja Ganu, Monojit Choudhury, and Sunayana Sitaram. 2021. GCM: A toolkit for generating synthetic code-mixed text. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 205–211, Online. Association for Computational Linguistics.

Kushagra Singh, Indira Sen, and Ponnurangam Kumaraguru. 2018. A Twitter corpus for Hindi-English code mixed POS tagging. In *Proceedings of the Sixth International Workshop on Natural Language Processing for Social Media*, pages 12–17, Melbourne, Australia. Association for Computational Linguistics.

Thamar Solorio, Shuguang Chen, Alan W. Black, Mona Diab, Sunayana Sitaram, Victor Soto, Emre Yilmaz, and Anirudh Srinivasan, editors. 2021. *Proceedings of the Fifth Workshop on Computational Approaches to Linguistic Code-Switching*. Association for Computational Linguistics, Online.

Victor Soto and Julia Hirschberg. 2017. Crowdsourcing universal part-of-speech tags for code-switching. In *Proc. Interspeech 2017*, pages 77–81.

Ishan Tarunesh, Syamantak Kumar, and Preethi Jyothi. 2021. From machine translation to code-switching: Generating high-quality code-switched text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3154–3169, Online. Association for Computational Linguistics.

Genta Indra Winata, Andrea Madotto, Chien-Sheng Wu, and Pascale Fung. 2019. Code-switched language models using neural based synthetic data from parallel sentences. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 271–280, Hong Kong, China. Association for Computational Linguistics.

A Appendix

A.1 Qualification Test

Table 6 presents the questions used in the MTurk qualification test.

Word-level language identification
buenas noches **a** toda my people
bonne soirée everyone
la declaration no es correcto
não canso de escutar see you again
federer **champion** por octava time en halle
the best place in the world no **hay** otro
todo lo que quiero **son** texts como este
espero que esto **no** sea verdad
ella siempre toma mis emotions **tan** seriamente
this is das **perfekt** moment
Naturalness evaluation
charlie sheen write defiant abierta letter to those que tried chantajearlo (*unnatural*)
hoy i have one de those days que hate al world entero (*unnatural*)
literalmente all i do is soñar sobre scenarios imposibles (*natural*)
merece perder el campeonato because of his attitude (*natural*)
tú made good música and probably tocaste the vidas of muchos children (*unnatural*)

Table 6: Qualification test used on Mechanical Turk. In the word-level language identification task, we ask the crowd workers to identify the language of the highlighted token. In the naturalness evaluation, we ask the workers to mark the naturalness of the code-mixed sentences.