Code-Switched Language Models Using Neural Based Synthetic Data from Parallel Structures

Genta Indra Winata, Andrea Madotto, Chien-Sheng Wu, Pascale Fung

Kaili Vesik

UBC: Deep Learning for NLP Reading Group

Thursday, October 10, 2019

Overview

Code-switching in Linguistics

- Code switching: speaking or writing in one language and switching to another within the same sentence.
- Linguists have proposed constraints to generalize code-switching but it is difficult to account for syntactically different languages.
- Equivalence Constraint (EC) theory: code switches only occur where the sentence elements are normally ordered in the same way in each language.

Code-switching in Machine Learning

- Building a LM and an ASR to cope with intra-sentential code-switching is challenging due to unpredictability of code-switching points and data scarcity.
- Creating a large-scale dataset is very expensive, so code-switched data augmentation would be advantageous.

Existing Methods & Prior Work

- Using EC alone to generate code-switching sentences [Li and Fung, 2012] [Pratapa et al., 2018].
  - Potential performance issues due to erroneous results from word aligner and POS tagger.
- Generating synthetic code-switching sentences using SeqGAN-based model [Garg et al., 2018].
  - Underperforming results due to distribution being very different from real data.

Proposed Method for Data Augmentation

Goals:

- Language-agnostic code-switching data generation using Pointer-Gen network.
  - Learn code-switching constraints from small initial dataset; apply to both languages.
  - Copy mechanism uses words from parallel monolingual sentences by aligning and reordering word positions to form a grammatical code-switched sentence.
- Apply EC to languages with significantly different syntactic structures (e.g., English vs Mandarin Chinese).
- Remove dependency on aligner/tagger.
- Generate new sentences with similar distribution to original dataset.

Pointer-Gen

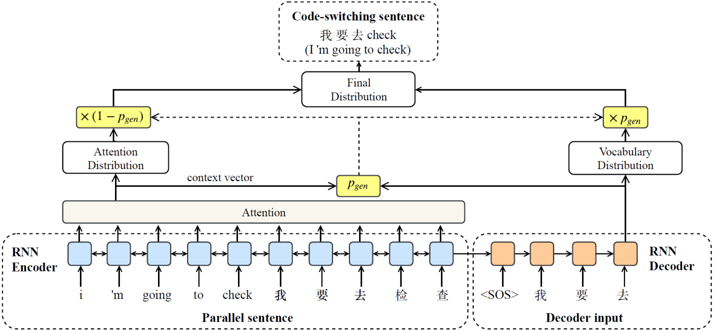
In general:

- Sequence-to-sequence system that incorporates both abstraction (summary/paraphrasing) and extraction (copying from source) to generate output, in order to ensure high-quality generation [See, 2017].
- Especially effective when output sequence *must* contain elements from input sequence (such as in code-switching).

For code-switching:

- Leverages parallel monolingual sentences to generate code-switching sentences.
- Trained on concatenated sequences of parallel sentences, constrained by code-switching texts.

Pointer-Gen for Code-Switching



- Input words are fed into BiLSTM encoder, which produces hidden state h_t in each step t.
- Decoder is a LSTM, which receives word embedding of previous word.

Pointer-Gen for Code-Switching

- Standard attention distribution a_t considers all encoder hidden states to derive context vector h_t^*.
- Vocabulary distribution $P_{voc}(w)$ is calculated by concatenating decoder state s_t and h_t^*.
- Generation probability $p_{gen} \in [0, 1]$ weights word generation from vocabulary vs copying from source text.

$$p_{gen} = \sigma\left(w_{h^*}^T h_t^* + w_s^T s_t + w_x^T x_t + b_{ptr}\right)$$

  where w_{h^*}, w_s, and w_x are trainable parameters and b_{ptr} is the scalar bias.
- Final distribution $P(w)$ is calculated as:

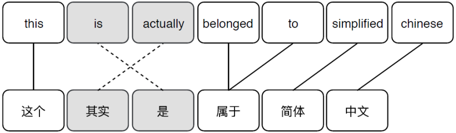
$$P(w) = p_{gen} P_{voc}(w) + (1 - p_{gen}) \sum_{i:w_i=w} a_i^t$$

Applying the Equivalence Constraint

- Code-switching only occurs where it does not violate the syntactic rules of either language.
- English and Mandarin have very different structures, so using a constituency parser will produce erroneous results.
- Instead, simplify sentences into linear structure and permit lexical substitution on non-crossing alignments between parallel sentences.

Applying the Equivalence Constraint

- During generation, permit any switches that do not violate the constraint.



Permissible switching

这个 其实 是 belonged to 简体 中文

这个 其实 是 belonged to simplified chinese

Impermissible switching

this 是 其实 belonged to simplified chinese

Experiment Data

Code-switching speech data:

- SEAME Phase II: a conversational English-Mandarin code-switching speech corpus consisting of spontaneous interviews and conversations.

Monolingual speech data:

- For Mandarin Chinese: HKUST, recordings of spontaneous telephone speech.
- For English: Common Voice, open-accented data collected by Mozilla.

Data Generation

1. Generate L_1 and L_2 sentences by using Google NMT to translate training set into both English and Chinese.
2. Use the parallel sentences to generate new code-switching sentences, tripling size of available data.
3. Complexity measured with:
    - Switch-Point Fraction (SPF): number of language switch-points in a sentence, divided by number of word boundaries in the sentence.
    - Code Mixing Index (CMI): number of non-matrix-language words in a sentence, divided by total number of words in the sentence, averaged over all sentences in corpus.

Test Effectiveness of Proposed Generation Method

Build a transformer-based end-to-end code-switching Automatic Speech Recognition (ASR) system:

- Begin with a model pretrained from monolingual speech, then jointly train speech from both languages to avoid the catastrophic forgetting that arises when training one after the other.
- Compare results of training on data generated by various methods:
  - Real data
  - Data augmented by Equivalence Constraint only
  - Data augmented by SeqGAN
  - Data augmented by Pointer-Gen (with EC as a substep)

Comparison of real code-switching data to generated data:

	Train	**Dev**	**Test**
# Speakers	138	8	8
# Duration (hr)	100.58	5.56	5.25
# Utterances	90,177	5,722	4,654
# Tokens	1.2M	65K	60K
CMI	0.18	0.22	0.19
SPF	0.15	0.19	0.17

	EC	**SeqGAN**	**Pointer-Gen**
# Utterances	270,531	270,531	270,531
# Words	3,040,202	2,981,078	2,922,941
new unigram	13.63%	34.67%	4.67%
new bigram	69.43%	80.33%	46.57%
new trigram	99.73%	141.56%	69.38%
new four-gram	121.04%	182.89%	85.07%
CMI	0.25	0.13	0.25
SPF	0.17	0.2	0.17

Training Strategies

1. Baseline: train on real code-switching data only (rCS).
2. Train on augmented data only:
   a. EC
   b. SeqGAN
   c. Pointer-Gen
3. Train on augmented data concatenated with rCS:
   a. EC & rCS
   b. SeqGAN & rCS
   c. Pointer-Gen & rCS
4. Two-step training: first only with augmented data, then fine-tuning with rCS:
   a. EC $\rightarrow$ rCS
   b. SeqGAN $\rightarrow$ rCS
   c. Pointer-Gen $\rightarrow$ rCS

Hypotheses

- Results from training on 2a and 2b (only EC and only SeqGAN) will not be as good as baseline 1 (rCS).
- Results from training on 3a and 3b (EC & rCS and SeqGAN & rCS) will outperform baseline.
- Result from training on 2c (Pointer-Gen only) will be on par with baseline, since Pointer-Gen is learning patterns from rCS and generates sequences with similar code-switching points.

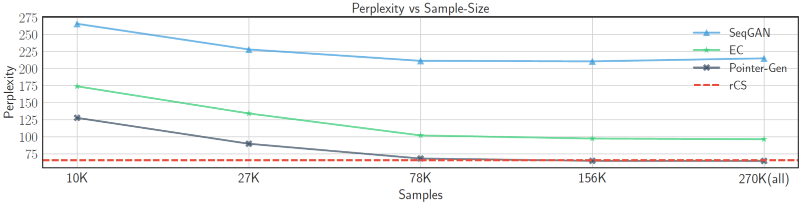
Language Model Performance

Token-level perplexity (PPL):

Training Strategy	Overall		en-zh		zh-en		en-en		zh-zh	
	valid	test	valid	test	valid	test	valid	test	valid	test
Only real code-switching data										
(1) rCS	72.89	65.71	7411.42	7857.75	120.41	130.21	29.31	29.61	244.88	246.71
Only generated data										
(2a) EC	115.98	96.54	32865.62	30580.89	107.22	109.10	28.24	28.2	1893.77	1971.68
(2b) SeqGAN	252.86	215.17	33719	37119.9	174.2	187.5	91.07	88	1799.74	1783.71
(2c) Pointer-Gen	72.78	64.67	7055.59	7473.68	119.56	133.39	27.77	27.67	234.16	235.34
Concatenate generated data with real code-switching data										
(3a) EC & rCS	70.33	62.43	8955.79	9093.01	130.92	139.06	26.49	26.28	227.57	242.30
(3b) SeqGAN & rCS	77.37	69.58	8477.44	9350.73	134.27	143.41	30.64	30.81	260.89	264.28
(3c) Pointer-Gen & rCS	68.49	61.57	7146.08	7667.82	127.50	139.06	26.75	26.96	218.27	226.60
Pretrain with generated data and fine-tune with real code-switching data										
(4a) EC → rCS	68.46	61.42	8200.78	8517.29	101.15	107.77	25.49	25.78	247.3	258.95
(4b) SeqGAN → rCS	70.61	64.03	6950.02	7694.2	114.82	122.84	28.5	28.73	236.94	244.62
(4c) Pointer-Gen → rCS	**66.08**	**59.74**	**6620.76**	**7172.42**	**114.53**	**127.12**	**26.36**	**26.40**	**216.02**	**222.49**

Language Model Performance

Effect of data size:



ASR Performance

Character Error Rate (CER):

- Determine overall CER as well as individual CERs for Mandarin Chinese and English.
- Calculates distance of two sequences as the *Levenshtein Distance*, "the minimum number of single-character edits (insertions, deletions or substitutions) required to change one sequence into the other" ["Levenshtein distance," 2019].

ASR Performance

Character Error Rate (CER):

Model	Overall	en	zh
Baseline	34.40%	41.79%	35.94%
+ Pre-training	32.76%	40.06%	32.44%
+ LM (rCS)	32.25%	39.45%	31.90%
+ LM (Pointer-Gen $\rightarrow$ rCS)	**31.07%**	**38.39%**	**30.85%**

Interpretability

Visualization of Pointer-Gen's attention weights:

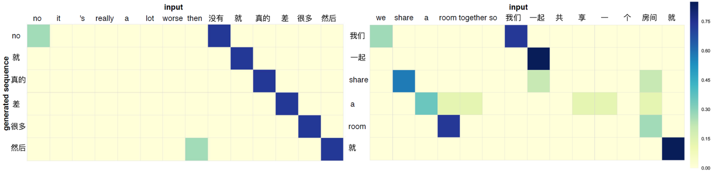


Figure 4: The visualization of pointer-generator attention weights on input words in each time-step during the inference time. The y-axis indicates the generated sequence, and the x-axis indicates the word input. In this figure, we show the code-switching points when our model attends to words in the L_1 and L_2 sentences: **left:** ("no","没有") and ("then","然后"), **right:** ("we","我们"), ("share", "一起") and ("room","房间").

- Attention weights show that the model can identify code-switching points, word alignments, and translations without begin given explicit information.

Code-Switching Patterns

Most common POS tags that trigger code-switching:

rCS		Pointer-Gen		
POS tags	**ratio**	**POS tags**	**ratio**	**examples**
English				
NN (noun)	56.16%	NN (noun)	55.45%	那个 **consumer** 是不 (that consumer is not)
RB (adverb)	10.34%	RB (adverb)	10.14%	okay **so** 其实 (okay so its real)
JJ (adjective)	7.04%	JJ (adjective)	7.16%	我很 **jealous** 的每次 (i am very jealous every time)
VB (verb)	5.88%	VB (verb)	5.89%	**compared** 这个 (compared to this)
Chinese				
VV (other verbs)	23.77%	VV (other verbs)	23.72%	讲的要**用** microsoft word (i want to use microsoft word)
M (measure word)	16.83%	M (measure word)	16.49%	我们有这**个** god of war (we have this god of war)
DEG (associative)	9.12%	DEG (associative)	9.13%	我们**的** result (our result)
NN (common noun)	9.08%	NN (common noun)	8.93%	我应该不会讲**话** because intimidated by another (i shouldn't talk because intimidated by another)

Code-Switching Patterns

- Distribution of common code-switching points in Pointer-Gen data is similar to rCS, which indicates that this model can learn code-switching points.

Conclusion

- Pointer-Gen can bse used to generate synthetic code-switching sentences.
- The proposed language model can learn code-switching patterns without requiring any word alignments or consitutency parsers.
- Crucially, the model can be used even for languages that are syntactically different.
- Depending on training strategy, the model can outperform equivalence constraint based models.
- The model can be used to improve the performance of an end-to-end automatic speech reognition system.

References

Saurabh Garg, Tanmay Parekh, and Preethi Jyothi (2018)
Code-switched language models using dual rnns and same-source pretraining
Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 3078 – 3083.

Ying Li and Pascale Fung (2012)
Code-switch language model with inversion constraints for mixed language speech recognition
Proceedings of COLING 2012, 1671 – 1680.

Adithya Pratapa, Gayatri Bhat, Monojit Choudhury, Sunayana Sitaram, Sandipan Dandapat, and Kalika Bali (2012)
Language modeling for code-mixing: The role of linguistic theory based synthetic data
Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) 1, 1543 – 1553.

References continued

Abigail See (2017)
"Taming Recurrent Neural Networks for Better Summarization"
Abigail See, 16 April 2017, www.abigailsee.com/2017/04/16/taming-rnns-for-better-summarization.html

Genta Indra Winata, Andrea Madotto, Chien-Sheng Wu, Pascale Fung (2019)
Code-Switched Language Models Using Neural Based Synthetic Data from Parallel Sentences.

"Levenshtein distance."
Wikipedia: The Free Encyclopedia. Wikimedia Foundation, Inc. 11 September 2019. Web. 8 October 2019, en.wikipedia.org/wiki/Levenshtein_distance