U dunno meh? Generating Singlish Messages

Yuzhou Yan
McGill University

Yuzhou Guo
McGill University

Cindy Wang
McGill University

Abstract

Computational processing of text messages can be a challenging problem when we want generated texts to make sense and display a certain degree of creativity. In the domain of Singapore English specifically, we trained three models, including Markov Chain, Bidirectional-LSTM, RNN model with character tokenization and RNN model with word tokenizing, to auto-generate Singlish-like text messages. Out of all models, the RNN model with word tokens performs the best from human evaluators' perspective and copy long phrases from the corpus less frequently.

1 Introduction

Automatic generation of text that is close to the quality of human-generated text has a lot of applications, such as language translation, chatbots, answering customer queries, etc. However, it might be a challenge to process text messages due to the use of contractions, unconventional expressions and oftentimes misspelling; the style of text messaging may also differ in different English-speaking regions. Genre-wise, current work on text generation focus mostly on more literary and formal styles, such as lyrics (Godinez, 2017), poems (Krivitski, 2018) and customer reviews (Maqsud, 2015), and a majority of such studies in English are based on texts from major anglophone regions like the UK and North America. We hope to investigate whether the state-of-the-art techniques are equally applicable to Singapore vernacular English, which is understudied or under-recorded both in terms of its genre type and as a less popular variety of English.

Singapore colloquial English (better known as Singlish) coexists with Standard Singaporean English (SSE) in Singapore, a city state with 76.2% ethnic Chinese, 15.0% Malays and 7.4% ethnic Indians. While SSE is virtually identical to Standard British English, Singlish is instead a creole with English grammar and vocabulary as its backbone but receives minor influence from Malay, Tamil, and more importantly, both lexical and syntactical influence from various Chinese dialects (Tien, 2010). Text messages closely resemble daily conversation and is suitable as a near representation of Singlish.

An example taken from *The National University of Singapore SMS Corpus* (Chen and Kan, 2015) is as follows:

> "Oh, he start lect now liao.. Wah, wait 4 who ah?"

In plain English, this means "Oh, he already started the lecture. Wow, who was he waiting for?"

wah is an interjection from Chinese; *liao* is a Chinese aspect marker for the completion of an action; and *ah* is a sentence-final particle signifying a soft question. In addition, the text also shows heavy influence from analytical features of Chinese grammar in the loss of inflections and change in word order.

Our manual part-of-speech (POS) tagging on the text messages (Table 1) further illustrates the distinction of Singlish from other English varieties. The backbone of our POS tags is Penn Treebank, and we added categories for Chinese-origin words as well as contractions. Although the POS tagging effort was an unfinished attempt to train a different model for our central hypothesis, it nevertheless effectively demonstrates how Singlish receives influence from Chinese.

We use three different models - Markov Chain, Bidirectional Long Short-term Memory (Bi-LSTM), and Recurrent Neural Network (RNN) -

Tags from Chinese	Count	%
Total # of tagged words	3895	100%
CHY - sentence-final particles	78	2.00%
CHU - auxiliary words	23	0.59%
CHN - noun	16	0.41%
CHI - interjection	15	0.39%
CHV - verb	6	0.15%
CHO - others	5	0.13%

Table 1: Counting tags for words that find their roots in varieties of Chinese languages. Categories are adapted from Yu and Zhu (2017). CHY, the most attested POS, includes sentence-final particles common in Singlish, such as *lah*, *lor*, and *meh*.

to learn from the pre-processed corpus and then generate Singlish text messages, with the goal of determining which model gives messages that makes the most sense, resembles Singlish, and is the most creative. We hypothesize that RNN will produce messages that best achieve the first two goals, given its ability to learn long-term dependencies and therefore better model grammatical dependencies and flow in meaning. It is also more likely to produce original messages different from those in the corpus.

2 Related Work

Text generation has been a popular field of exploration since the advent of modern techniques in Natural Language Processing. While classical machine learning models like Markov Chain have not grown out of date and are still being used by Maqsud (2015) and J. (2018) in text generation, the rise of multiple techniques employing neural network has made the world of NLP research gravitate towards them. For example, Krivitski (2018) used recurrent neural network to generate poems in English, German, and Russian, and Godinez (2018) employed Long Short-Term Memory models to generate poems with the style of a particular poet.

One study very relevant to our project on Singlish is the effort by Yip (2018) to use LSTM to generate Singlish text messages. His work is inspirational in the sense of paying attention to this understudied variety of English, and his result by using LSTM model is also a good starting point based on which our study will improve further.

3 Methodology

The dataset we will be using for this work is The National University of Singapore SMS Corpus ("the Corpus") (Chen and Kan, 2015). It contains over 54,000 mobile text massage records from English speakers in Singapore, mainly students at National University of Singapore, along with relevant information about the senders and recipients, such as their nationality and native language.

3.1 Text pre-processing

For dataset preparation, we exported the original text corpus, which is of xml format, into json and txt formats respectively to facilitate model training. With the json file, we first removed punctuations from the text and then tokenized each text using *text_to_word* from the Keras module to create a sequence length of 5 using the first 1,000 text messages for training the Bi-LSTM model and creating a vocabulary. Next, each word was encoded with an index, representing the vocabulary with one-hot vectors. As for the txt file, it is used for training the Markov Chain model and RNN. We removed punctuation in MC training but preserved it in RNN. We used all messages sent by Singaporeans (count: 20814) in the corpus for Markov Chain, but used only 3,000 text messages for best training of Recurrent Neural Network.

3.2 Markov Chain

We base our Markov Chain model on the one developed by J. (2018). A second-order MC model assumes that each word depends on two previous words only. It learns the probability distribution of the occurrence each word together with a pair of two previous words. We employed word-tokenization and character-tokenization, and trained our MC model with second, third and fourth order respectively. We then used a function to sample out from the distributions and generated words one by one, each based on its probability given the two previous words generated. We only included the results from second-order, word-tokenized MC model in our evaluation, because higher order models over-fit the corpus too much, and results from character-tokenized models contain numerous badly spelt words and are obviously undesirable.

3.3 Bidirectional-LSTM

We received inspiration from the previous work of Yip (2018) in setting up the Bi-LSTM model.We applied a word embedding layer of size 3 to map the pre-processed representation into a specified number of dimensions. The many-to-many architecture, along with the embedding layer, Bi-LSTM layer, dropout layer, and output layer, was applied with softmax as its activation function to find the most likely word in the vocabulary to use. We then fit the Bi-LSTM model using Adam Optimizer and used Sparse Categorical Cross Entropy as its loss function. Finally, we fed three start words (”I will be”) into the model to initiate the text generation process.

3.4 Recurrent Neural Network

We based the RNN model off a character-level architecture, which made use of attention-weighting and skip-embedding techniques to accelerate training and improving accuracy. However, for experimentation purposes, we trained the model at both character-level and word-level separately to compare the quality of generated Singlish text. The parameter *temperature*, which refers to the level of freedom of the text being generated, was set to 1.0 (maximum) for both model architectures.

4 Results

In this section, we display the evaluation results from a variety of model architectures. Traditional means such as BLEU and ROUGE, which evaluated generated texts based on parallel human work, is not applicable to our experiment. We instead sought to gauge creativity with cardinality and reasonableness with Turing-test style human evaluation.

4.1 Cardinality Count

We use cardinality to measure the originality of the generated texts. We defined two types of cardinality count: the cardinality relative to the corpus data and the cardinality within the output strings themselves. For both types, we count how many n-grams (n consecutive words) from the generated text find twins in the corpus or among the very same group of generated texts. We counted repetition in terms of three-grams, four-grams, five-grams and six-grams. In practice, we cut data in the cleaned corpus and generated texts into n-grams, and then ran the counting script and printed out the duplication percentages.

	N-gram	**D.C.%**	**D.S.%**
Markov Chain	3-gram	98.42	0.00
	4-gram	63.26	0.00
	5-gram	40.29	0.00
	6-gram	27.32	0.00
	N-gram	**D.C.%**	**D.S.%**
LSTM	3-gram	65.67	0.00
	4-gram	50.42	0.00
	5-gram	43.30	0.00
	6-gram	35.96	0.00
	N-gram	**D.C.%**	**D.S.%**
RNN - Character	3-gram	10.57	3.84
	4-gram	0.88	0.39
	5-gram	0.10	0.10
	6-gram	0.00	0.00
	N-gram	**D.C.%**	**D.S.%**
RNN - Word	3-gram	34.49	11.71
	4-gram	24.19	3.60
	5-gram	18.16	0.94
	6-gram	13.91	0.41

Table 2: Recording the duplication percentage of different models, both compared to corpus (20814 sentences) and compared to self (50 sentences each). For each model, duplication percentage was recorded separately for various n-gram choice. D.C. stands for *Duplication compared to Corpus* and D.S. stands for *Duplication compared to Self*.

The result of duplication percentage (Table 2) shows that for each model, the duplication percentage either to corpus or to itself decreases as we compare longer ”n-grams”. That is, the more consecutive words we considered while counting, the less frequently duplication rises.

It is also found that Markov Chain model has the highest duplication percentage to corpus, mostly likely due to its property of word chaining since the next word generated highly depends on the previous two words which all come from the corpus. The LSTM model also has a high duplication percentage to the corpus, starting off lower than the Markov Chain model (65.67%) but the percentage decreases more slowly along the increment of number of grams considered. It is also worth noticing that both the Markov Chain and the LSTM model have zero duplication compared to self.

For the RNN models, they both have a very low duplication percentage to self output data. The one with characters as tokens has exceptionally low duplication percentages in general since it learnt from the combination of characters rather than entire words, and the one with words have a relatively low duplication percentage with corpus compared to the other two models. More analysis about the cardinality count and duplication result will be discussed in the subsection *Diccussion*.

4.2 Human Assessment

We adapted Ahamad (2019)'s human evaluation method, where evaluators are asked to give a score for every sentence based on how likely they think the sentence is written by a particular author. In our evaluation, we invited 22 undergraduate students mostly from Singapore who have spent at least four years in an educational institution in Singapore and were in Singapore for at least 6 months in the past year. We created two different groups of sentences for them to assess. For each group, we randomly chose 12 sentences from each model (Markov Chain, Bi-LSTM, RNN character-tokenized, RNN word-tokenized) as well as 21 from the Singlish corpus, keeping with the standard that $\frac{1}{3}$ from each model are between the length of 3 to 6 words, $\frac{1}{3}$ between 7 and 12 and $\frac{1}{3}$ longer than 13. Note that 6 and 12 are 3-quantiles of sentence lengths in the corpus.

We presented evaluators with the message below, followed by 69 sentences.

> Dear evaluator,
>
> In the 69 sentences below, there are some real SMS messages collected by the National University of Singapore, and there are also machine-generated SMS messages. Your task will be to give a score (from 1 to 5) to each message below.
>
> 1 means the message sounds the most fake; 5 means the message is totally real.
>
> Enter your score within the rightmost "[]" brackets in each line. No need to scrutinize too much; just follow your gut feeling. Happy evaluating!

After receiving the survey results from our evaluators, we calculated the mean value and standard deviation of scores (on a 1-5 scale) for each model as well as the original Singlish SMS corpus. Then, we calculated overall averages of scores from both group one and group two.

	MC	**LSTM**	**RNNc**	**NUS**	**RNNw**
G1 Mean	3.167	3.488	2.503	4.082	3.576
G1 S.D.	0.693	0.683	1.071	0.526	0.500
G1 N.S.E.	132	131	131	231	129
G2 Mean	3.424	2.901	2.209	3.832	3.242
G2 S.D.	0.457	0.510	0.698	0.423	0.560
G2 N.S.E.	132	121	131	208	132
Mean	3.295	3.206	2.356	3.964	3.408

Table 3: Giving two groups of sentences for evaluation and record a number of values for each model as well as the original corpus data from NUS. S.D. stands for *Standard Deviation* and N.S.E. stands for *Number of Sentences Evaluated*. RNNc stands for *RNN model using character tokens*. RNNw stands for *RNN model using word tokens*. Mean refers to the overall mean from both group one and group two.

The human assessment result (Table 3) demonstrates that out of the overall score of 5, the corpus sentences from NUS gets the highest score of 3.964 with no doubt with the smallest standard deviation.

With the NUS score as a reference, the RNN model with word tokens performs the best out of all models with a mean value of 3.408, and the RNN model using character tokens performs the worst with a score of 2.356 with largest standard deviation of all. This means that, from the human evaluation perspective, using the char-tokenized RNN model generates broken Singlish texts in general but still some good results here and there. Looking at the overall mean score, the Markov Chain model and LSTM performs roughly the same with the score of 3.295 and 3.206.

	MC	**LSTM**	**RNNc**	**NUS**	**RNNw**
G1 Mean	2.417	2.500	1.364	3.714	3.091
G1 N.S.E.	12	12	11	21	11

Table 4: Survey result from Daniel Li: a linguistics major with extensive experience in researching colloquial Singapore English at Nanyang Technological University.

It is worth mentioning that one of our evaluators Daniel Li specializes in Singlish research.

His evaluation result (Table 4) matches the general feedback from all evaluators, albeit with lower scores in general.

5 Discussion and conclusion

In conclusion, the RNN model with word tokens performs the best to accomplish the task of generating Singlish-like text messages. Human evaluators generally give high scores to sentences produced by this model. It outperforms its character-tokenization counterpart because it learn each word as a whole and has no risk of churning out misspelt words.

The RNN model with character tokens, also the only model uses character as tokens, performs the worst. Since a word can be expressed in many ways in a Singapore daily text message, such as writing *don't know* as *dunno*, *dunnoe*, *donno*, or *dun know*, it will be tough to use recurrent models on characters to capture so many more different character sequences.

RNNw also outperforms Bi-LSTM and Markov Chain model. Following the word-level RNN model, the Markov Chain performs the second best. Compared to MC, RNNw has obvious advantages such as the ability to learn and retain long-term dependencies, and to generate each next word conditioned on the entire history of previous words generated. In contrast, Markov Chain only keeps track of probabilities in a fixed window–it does not look farther than two words before when generating each next word. Therefore, the Markov Chain model is less likely to convince our human assessors because its sentences flow less naturally and sometimes looks like gluing irrelevant ideas together.

Despite its capacity to consider long-distance dependencies and even to conquer RNN's problem of vanishing gradient, LSTM is not performing well in our experiment. Its mean score comes lower than those of RNNw and Markov Chain. One reason lies in the output format of LSTM: it generates texts in a continuous stream rather than discrete sentences with starting and ending points. To prevent bias in evaluation, when we extract sentences for evaluation, we had to cut the stream only at every 5, 10, or 15th word, rather than drawing sentence boundaries with our judgment. This creates an inevitable shortcoming in evaluation, because some sentence beginning and endings may look abnormal; scores for LSTM turns out lower than expected. In the future, we will have to investigate how to adapt the output format of LSTM text generators such that they are comparable to those from other text generation models.

If we look at the duplication percentage at the six-gram level, the output text messages from RNN word model can be considered rather "creative" and "unique" in a general sense, since the output that the model copies from the corpus is under a reasonable level (about 14%). For this, we need to thank the model parameter *temperature* which was set to maximum. The maximal level of freedom was applied to the selection and sampling of every next unit. This brings us great performance in terms of originality. Also, RNN focuses less on short phrases but more on high-level coherence–this also helps reduce direct copying.

Now that we have realized the capacity and strengths of each model, we also look into possible further improvements. For the LSTM model, the output is expected to be better since we can control the states of the network more flexibly through the LSTM nodes. In future explorations, rather than using the embedding layer of a size three, a larger size can be implemented to increment the dimension size in order to improve the overall performance. The Markov Chain model might be improved by using higher order models so that next words can be generated depending on a long sequence of previous words, and results will be more recognized by Singlish speakers; however, such models will also face more severe challenges in repetition and copying. In general, recurrent neural networks turned out to be a better approach as the number of states that needs to be memorized is relatively large, which doesn't fall under the constraints of simple chaining. In addition, as neural networks tend to capture language variations and are often explored as the new gold standard in language generation, RNN became a fundamental architecture for sequences of any kind, and in this work specifically, we have demonstrated its potential in modelling Singlish texts.

Statement of contributions

Yuzhou Yan researched on Singlish, researched and conducted experiments with the Markov Chain model, conducted manual part-of-speech tagging, organized human evaluation, and is a main contributor in composing and editing the report.

Yuzhou Guo researched on evaluation methods, carried out machine evaluation and is a main contributor in writing and editing the report.

Cindy Wang worked on Bi-LSTM and Recurrent Neural Network (char-level and word-level), built and carried out experiments with these two models, as well as delivered corresponding methodologies, results, and discussions in detail.

References

Afroz Ahamad. 2019. Generating text through adversarial training using skip-thought vectors. In *2019 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 53–60, Minneapolis, Minnesota.

T. Chen and Min-Yen Kan. 2015. The national university of singapore sms corpus. ScholarBank@NUS Repository. [Dataset]. https://doi.org/10.25540/WVM0-4RNX.

Josue Espinosa Godinez. 2017. The accuracy of recurrent neural networks for lyric generation. *Conference'17*.

Josue Espinosa Godinez. 2018. Semantic-map-based assistant for creative text generation.

Ashwin M. J. 2018. Next word prediction using markov model. *Medium.com*.

Denis Krivitski. 2018. Generation of poems with a recurrent neural network. *Medium.com*.

Umar Maqsud. 2015. Synthetic text generation for sentiment analysis. In *6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 156–161, Lisboa, Portugal.

Adrian Tien. 2010. Chinese-based lexicon in singapore english, and singapore-chinese culture. ScholarBank@NUS Repository. [Dataset]. https://doi.org/10.25540/WVM0-4RNX.

Jason Yip. 2018. Generating singlish text messages with a lstm network. *Medium.com*.

Shiwen Yu and Xuefeng Zhu. 2017. Contemporary chinese grammar information dictionary. [Dataset]. http://dx.doi.org/10.18170/DVN/EDQWIL.

A Appendices

A.1 Examples of POS tagging

Bugis/NNP oso/RB near/IN wat/CHY .../:
Oh/UH oh/UH .../: Wasted/VBN .../: Den/RB muz/MD chiong/CHV on/IN sat/NN n/CC sun/NN liao/CHU .../:
U/PRP meet/VBP other/JJ fren/NN dun/DUN wan/VB meet/VB me/PRP ah/CHY .../: Muz/MD b/VB a/DT guy/NN rite/JJ .../:

A.2 Examples of generated text messages from different models

Markov Chain
lol wear specs siboooo
yo yo why you need to buy tickets for lord of the stick is translating or not pretty haha
k thanks u sold them off successfully

Bi-LSTM (an uninterrupted stream)
i will be late ah meet you at 1pm orchard to do u coming to school ah so u have ur need to buy lunch for me i eat maggi first den decide later i juz buy when we get

RNN: character-tokenized
Ok... Thk u wanna go for comf stupide only i come too a bus busy?
you finish tmr...
Yup of us tomor today... U went all lor... U want to go still done where time i think u in twice at home on Nm ü mah... come to just still have to come do u?

RNN: word-tokenized
u reaching serangoon ?
ll b goin casual lor . . . u workin dinner ?
ya , my tai tai life gona end in a month time . . . sob . . . old bird ? haha . . . den wat u doing ? having ur dinner ?

A.3 Examples of real Singlish messages

Oh... Kay... On sat right?
I need... Coz i never go before
I also nt going...Tink u better tell cow...Hehe
U cooking tonight?
I'm gona b late too...
No. I'll be having a proj meeting.
Sree, if u're too tied up with other projects, rest assured that wecan follow up with the biz law assignment, just let us know what isleft so that

we’ve enough response time. I’m sorry to pester u again!Haha.
It’s okay. Go see doctor lah. The meeting nvm de lah. Okay will meetnear a toilet!