Controlling Sequence-to-Sequence Models - A Demonstration on Neural-based Acrostic Generator

Liang-Hsin Shen, Pei-Lun Tai, Chao-Chung Wu, Shou-De Lin
Department of Computer Science and Information Engineering,
National Taiwan University
{laurice, b04902105, r05922042, sdlin}@csie.ntu.edu.tw

Abstract

An acrostic is a form of writing for which the first token of each line (or other recurring features in the text) forms a meaningful sequence. In this paper we present a generalized acrostic generation system that can hide certain message in a flexible pattern specified by the users. Different from previous works that focus on rule-based solutions, here we adopt a neural-based sequence-to-sequence model to achieve this goal. Besides acrostic, users can also specify the rhyme and length of the output sequences. To the best of our knowledge, this is the first neural-based natural language generation system that demonstrates the capability of performing micro-level control over output sentences.

1 Introduction

Acrostic is a form of writing aiming at hiding messages in text, often used in sarcasm or to deliver private information. In previous works, English acrostic have been generated by searching for paraphrases in WordNet's synsets (Stein et al., 2014). Synonyms that contain needed characters replace the corresponding words in the context to generate the acrostic. Nowadays Seq2Seq models have become a popular choice for text generation, including generating text from table (Liu et al., 2018), summaries (Nallapati et al., 2016), short-text conversations (Shang et al., 2015), machine translation (Bahdanau et al., 2015; Sutskever et al., 2014) and so on. In contrast to a rule-based or template-based generator, such Seq2Seq solutions are often considered more general and creative, as they do not rely heavily on pre-requisite knowledge or patterns to produce meaningful content. Although several works have presented automatic generation on rhymed text (Zhang and Lapata, 2014; Ghazvininejad et al., 2016), the works do not focus on controlling the rhyme of the generated content. One drawback of a neural-based Seq2Seq model is that the outputs are hard to control since the generation follows certain non-deterministic probabilistic model (or language model), which makes it non-trivial to impose a hard-constraint such as acrostic (i.e. micro-controlling the position of a specific token) and rhyme. In this work, we present an NLG system that allows the users to micro-control the generation of a Seq2Seq model without any post-processing. Besides specifying the tokens and their corresponding locations for acrostic, our model allows the users to choose the rhyme and length of the generated lines. We show that with simple adjustment, a Seq2Seq model such as the Transformer (Vaswani et al., 2017) can be trained to control the generation of the text. Our demo system focuses on Chinese and English lyrics, which can be regarded as a writing style in between articles and poetry. We consider a general version of acrostic writing, which means the users can arbitrarily choose the position to place acrostic tokens. The 2-minute demonstration video can be found at https://youtu.be/9tX6ELCNMCE.

2 Model Description

Normally a neural-based Seq2Seq model is learned using input/output sequences as training pairs (Nallapati et al., 2016; Cho et al., 2014a). By providing sufficient amount of such training pairs, it is expected that the model learns how to produce the output sequences based on the inputs. Here we would like to first report a finding that a Seq2Seq model is capable of discovering the hidden associations between inputting $control\ signals$ and outputting sequences. Based on the finding we have created a demo system to show that the users can indeed guide the outputs of a Seq2Seq model in a

Proceedings of the 2019 EMNLP and the 9th IJCNLP (System Demonstrations), pages 43–48
Hong Kong, China, November 3 – 7, 2019.

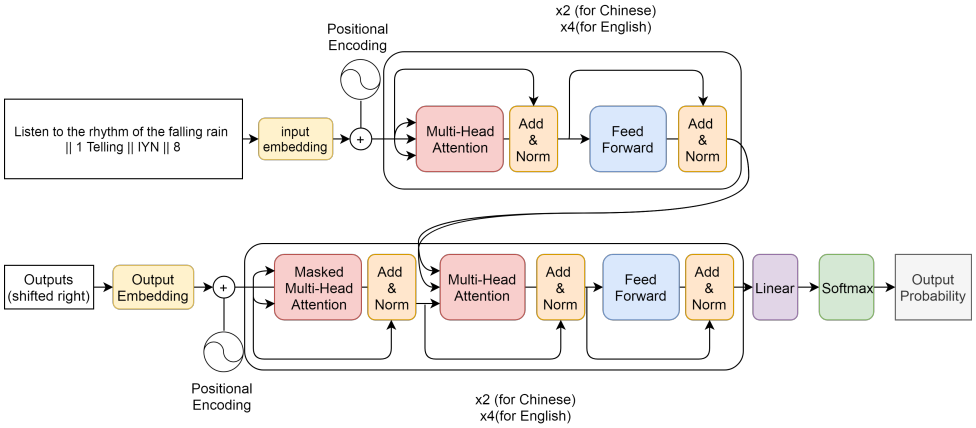

Figure 1: The structure of Transformer model.

fine-grained manner. In our demo, the users are allowed to control three aspects of the generated sequences: rhyme, sentence length and the positions of designated tokens. In other words, our Seq2Seq model not only is capable of generate next line satisfying the length and rhyme constraints provided by the user, it can also produce the exact word at a position specified by the user. The rhyme of a sentence is the last syllable of the last word in that sentence. The length of a sentence is the number of tokens in that sentence. To elaborate how our model is trained, we use three consecutive lines (denoted as S_1, S_2, S_3) of lyrics from the song "Rhythm of the Rain" as an example. Normally a Seq2Seq model is trained based on the following input/output pairs.

> s_1: Listen to the rhythm of the falling rain $\rightarrow$ s_2: Telling me just what a fool I've been
>
> s_2: Telling me just what a fool I've been $\rightarrow$ s_3: I wish that it would go and let me cry in vain

With some experiments on training Seq2Seq models, we have discovered an interesting fact: By appending the $control\ signals$ in the end of the input sequences, after seeing sufficient amount of such data, the Seq2Seq model can automatically discover the association between input signals and outputs. Once the associations are identified, then we can use the $control\ signals$ to guide the output of the model. For instance, here we append additional control information to the end of the training sequence as below

> s_1: Listen to the rhythm of the falling rain $\|$ 1 $Telling$ $\|$ IYN $\|$ 8 $\rightarrow$ s_2: Telling me just what a fool I've been
>
> s_2: Telling me just what a fool I've been $\|$ 2 $wish$ 6 go $\|$ EYN $\|$ 12 $\rightarrow$ s_3: I wish that it would go and let me cry in vain

The three types of $control\ signals$ are separated by "$\|$". The first $control\ signal$ indicates the position of the designated words. 1 $Telling$ tells the system the token $Telling$ should be produced at the first position of the output sequence s_2. Similarly, 2 $wish$ 6 go means that the second/sixth token in the output sequence shall be $wish/go$. The second $control\ signal$ is the rhyme of the target sentence. For instance, IHN corresponds to a specific rhyme (/In/) and EYN corresponds to another (/en/). Note that here we use the formal name of the rhyme (e.g. EYN) to improve readability. To train our system, any arbitrary symbol would work. The third part contains a digit (e.g. 8) to control the length of the output line.

By adding such additional information, Seq2Seq models can eventually learn the $meaning$ of the $control\ signal$ as they can produce outputs according to those signals with very high accuracy. Note that in our demo, all results are produced by our Seq2Seq model without any post-processing, nor do we provide any prerequisite knowledge about what length, rhyme or position really stands for to the model.

We train our system based on the Transformer model (Vaswani et al., 2017), though additional experiments show that other RNN-based Seq2Seq models such as the one based on GRU (Cho et al., 2014b) or LSTM would also work. The model consists of an encoder and a decoder. Our encoder

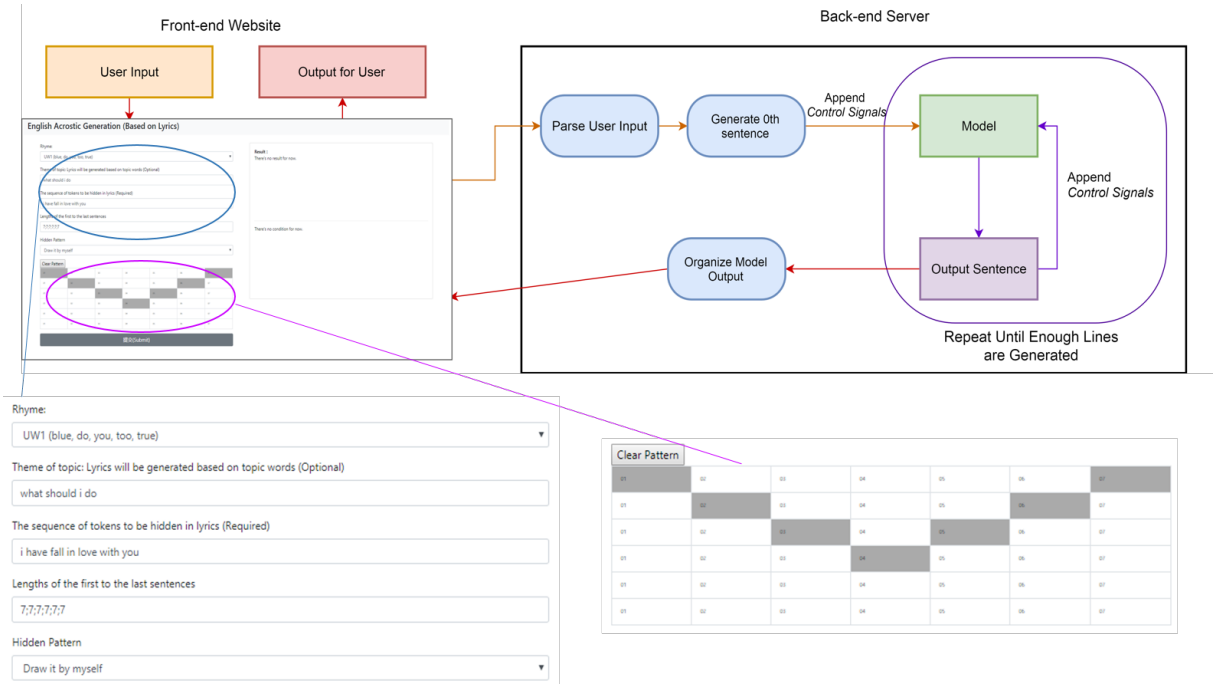

Figure 2: The structure of our acrostic generating system.

consists of two identical layers when training on Chinese lyrics and four identical layers when training on English lyrics. Each layer includes two sub-layers. The first is a multi-head attention layer and the second one is a fully connected feed-forward layer. Residual connections (He et al., 2016) are implemented between the sub-layers. The decoder also consists of two identical layers when training on Chinese lyrics and four identical layers when training on English lyrics.. Each layer includes three sub-layers: a masked multi-head attention layer, a multi-head attention layer that performs attention over the output of encoder and a fully-connected feed-forward layer. The model structure is shown in Figure 1. Note that in the original paper (Vaswani et al., 2017), Transformer consists of six identical layers for both encoder and decoder. To save resource, we start training with fewer layers than the original paper and discover that the model still performs well. Thus, we use fewer layers than the proposed Transformer model.

3 User Interface

Figure 2 illustrates the interface and data flow of our acrostic lyric generating system. First, there are several conditions (or *control signals*) that can be specified by the users:

- Rhyme: For Chinese lyrics, there are 33 different rhymes for users to choose from. As for English lyrics, there are 30 different rhymes for users to choose from.

- Theme of topic: The theme given by user is used to generate the zeroth sentence. In Chinese Acrostic demonstration, our system would pick a sentence from training set that is most similar to the user input, measured by the number of n-grams. As for English Acrostic demo, the user input of theme is directly used as the zeroth sentence.

- Length of each line: User can specify the length of every single line (separated by ;). For example, "5;6;7" means that the user wants to generate acrostic that contains 3 lines, with length equals to 5, 6, 7, respectively.

- The sequence of tokens to be hidden in the output sequences.

- Hidden Pattern: The exact positions for each token to be hidden. Apart from the common options, such as hiding in the first/last positions of each sentence or hiding in the diagonal positions, our system offers a more general and flexible way to define the pattern, realized through the *Draw It Myself* option. As shown in the bottom right corner of

Figure 2, a table based on the length of each line specified by the users is created for the users to select the positions to place acrostic tokens.

The generation is done on the server side. After receiving the *control signals* provided by users, the server first uses the given theme to search for a related line (denoted as *zeroth sequence*) from the lyric corpus, based on both sentence-level matching and character-level matching. Then the given condition of first sentence is appended to this *zeroth* sequence to serve as initial input to the Seq2Seq model for generating first line of outputs. Next, the given condition of second sentence is appended to the generated first line as input to generate the second line. The same process is repeated until all lines are generated.

4 Experiment and Results

4.1 Data set

We have two versions: one training on Chinese lyrics and one on English lyrics.

The Chinese lyrics are crawled from *Mojim lyrics site* and *NetEase Cloud*. To avoid rare characters, the vocabulary size is set to the most frequent 50,000 characters. The English lyrics are crawled from *Lyrics Freak*. The vocabulary size is set to the most frequent 50,000 words. For each line of lyrics, we first calculate its length and then retrieve the rhyme of the last token. To generate the training pairs, we randomly append to the input sequence *some tokens and their positions* of the targeting sequence as the first *control signal*, followed by the *rhyme* and then *length*. Below are two example training pairs:

s_1: Listen to the rhythm of the *falling rain* || 2 *me* 3 *just* || IYN || 8 $\rightarrow$ s_2: Telling me just what a fool I've been

s_2: Telling me just what a fool I've been || 2 *wish* 6 *go* 7 *and* || EYN || 12 $\rightarrow$ s_3: I wish that it would go and let me cry in vain

In total there are about 651,339/1,000,000 training pairs we use to train our Chinese/English acrostic systems.

4.2 Evaluation

Our system has three controllable conditions on generating acrostic: the positions of designated tokens, the rhyme of each line and the length of each line. The evaluation set consists of 30,000 lines that are not included in training data. We first evaluate how accurate the control conditions can be satisfied. As shown in Table 1, the model can almost perfectly satisfy the request from users. We also evaluate the quality of learned language model for Chinese/English lyrics. The bi-gram perplexity of original training corpus is 54.56/53.2. The bi-gram perplexity of generated lyrics becomes lower (42.33/42.34), which indicates the language model does learn a better way to represent the lyrics data. In this experiment we find that training on English lyrics is harder than training on Chinese lyrics. English has strict grammatical rules while Chinese lyrics have more freedom in forming a sentence. We also observe that the model tends to generate sentences that use the same words that appear in their previous sentences. This behavior might be learned from the repetition of lyrics lines.

4.3 Demonstration of Results

We provide our system outputs from different aspects.

The first example in Figure 3 shows that we can control the length of each line to produce a *triangle*-shaped lyrics.

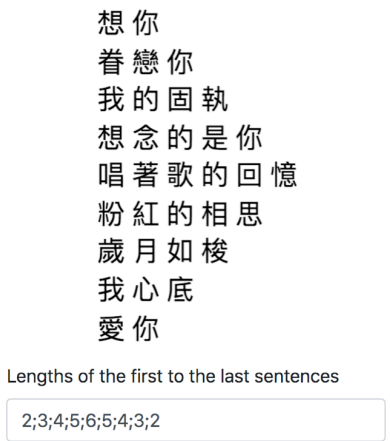

Figure 3: Length control of each sentence.

Second, we would like to demonstrate the results in generating acrostic. Some people use acrostic to hide message that has no resemblance with the content of the full text. We would show both English and Chinese examples generated by our system.

Figure 4 shows hiding a sentence in the first

Condition	Accuracy(Chinese)	Accuracy(English)
Character (CH) / Word (EN) Position	99.38%	98.21%
Rhyme	99.31%	97.67%
Sentence Length	99.90%	99.85%
Source	Perplexity(Chinese)	Perplexity(English)
Training data	54.56	53.2
Model generated data	42.33	42.34

Table 1: The accuracy of each condition tested on 30,000 lines and the perplexity of the original text and the text generated by our model.

word of each sentences. The sentence that being concealed in the lyrics is *I don't like you*, which is very different from the meaning of the full lyrics.

I said I want a real man
Don't you know that I really can
Like I really want to see you
You really want to be with me

Figure 4: Message in English lyrics: *I don't like you*.

甚至你不懂我的夢
怎麼我也不懂得等
一切都是因為女生
一切都可以因為夢
一切都可以化成風
我們都可以藏著夢

Rhyme:
eng (夢, 生, 風, 聲, 等)
Theme of topic: Lyrics will be generated based topic words
寬闊海洋
The sequence of tokens to be hided in lyrics
甚麼都可以藏
Lengths of the first to the last sentences
8;8;8;8;8;8
Hidden Pattern
Diagonal line

Figure 5: Hidden message in Chinese lyrics: 甚麼都可以藏 with rhyme *eng*.

Figure 5 shows a Chinese acrostic generated by our system. We hide a message 甚麼都可以藏 (Anything can be hidden) in the diagonal line of a piece of lyrics that talks about relationship and dream.

Third, we can also play with the visual shape of the designated words. Figure 6 shows an example

I	don't	want	to	be	a	part	of	you	anymore
We're	heading	for	the	big	world	outside	of	the	door
And	we'll	change	the	world	for	the	one	and	more
And	if	you	feel	like	I'm	in	love	forever	more
Then	again	I	wish	I'd	see	you	more	and	more
Then	I	would	learn	to	read	what	I	said	before

Figure 6: Message in English lyrics: *be the change you wish to see in the world*. To make the diamond shape clearer, the words are aligned.

of hiding a sentence in the shape of diamond in the generated lyrics. The message being concealed is *be the change you wish to see in the world*. Figure 7 shows that we can hide the message using the shape of a heart.

迴家路縱橫交錯在黃昏的時機
搖曳著斜斜影黃昏歲月的雨滴
落花雨水洗去疏狂心動的痕跡
搖曳著冷清的記憶浮現我心底
落花雨你淺淺的香味淡淡如昔
你是否也像幽暗一樣迷戀著你

Rhyme:
i (你, 裡, 起, 子, 己)
Theme of topic: Lyrics will be generated based on topic words (Optional)
The sequence of tokens to be hidden in lyrics (Required)
疏影橫斜水清淺暗香浮動月黃昏
Lengths of the first to the last sentences
13;13;13;13;13;13
Hidden Pattern
Draw it by myself

Figure 7: The designated characters form a heart. The sentence hidden in the lyrics is 疏影橫斜水清淺暗香浮動月黃昏 (The shadow reflects on the water and the fragrance drifts under the moon with the color of dusk) with rhyme *i*.

5 Conclusion

We show that by appending additional information in the training input sequences, it is possible to train a Seq2Seq model whose outputs can be controlled in a fine-grained level. This finding enables us to design and demonstrate a general acrostic generating system with various features controlled, including the length of each line, the rhyme of each line and the target tokens to be produced and their corresponding positions. Our results have shown that the proposed model not only is capable of generating meaningful content, it also follows the constraints with very high accuracy. We believe that this finding can further lead to other useful applications in natural language generation.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014a. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734.

Kyunghyun Cho, Bart van Merriënboer, Çalar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014b. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Marjan Ghazvininejad, Xing Shi, Yejin Choi, and Kevin Knight. 2016. Generating topical poetry. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1183–1191.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778.

Tianyu Liu, Kexiang Wang, Lei Sha, Baobao Chang, and Zhifang Sui. 2018. Table-to-text generation by structure-aware seq2seq learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 4881–4888.

Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Çaglar Gülçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pages 280–290.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1577–1586.

Benno Stein, Matthias Hagen, and Christof Bräutigam. 2014. Generating acrostics via paraphrasing and heuristic search. In *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014, Dublin, Ireland*, pages 2018–2029.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 6000–6010.

Xingxing Zhang and Mirella Lapata. 2014. Chinese poetry generation with recurrent neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 670–680.