國立臺灣大學電機資訊學院資訊工程學系
碩士論文
Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Taiwan University
Master Thesis

藏頭詩生成系統：Seq2Seq 控制訊號的應用
Acrostic Generating System: An Application of Control Signals on Sequence-to-Sequence Models

沈亮欣
Liang-Hsin Shen

指導教授：林守德博士
Advisor: Shou-De Lin, Ph.D.

中華民國 108 年 7 月
July, 2019

doi: 10.6342/NTU201902806

誌謝

首先，謝謝林守德老師一直以來的包容與指導，在我做研究的路上給了我非常多的幫助，再來謝謝和我一起做研究的戴培倫學弟，以及從大學就給我許多幫助的好夥伴李彥霆，最後謝謝這一路上陪伴我、幫助我的所有人們。

doi: 10.6342/NTU201902806

Acknowledgements

First, I would like to express my appreciation to my advisor, Prof. Shou-De Lin, who has given me a lot of advises on my research and has helped me a lot in finishing the thesis. Then, I would like to thank Pei-Lun Tai for being in my group and did the research with me. Also, I would like to thank Yen-Ting Lee for helping me since college time. Last but not the least, I appreciate the kindness of everyone that has helped me during my life.

doi: 10.6342/NTU201902806

摘要

藏頭詩是一種文體，指的是在一篇文章或詩詞的每句句首藏入能組成有意一詞組或句子的字，這篇論文中，將用更廣義的定義，將藏頭拓增成藏在任何位子，不管是斜線或圖形都是這篇裡「藏頭」所指的範圍。這篇論文中，我們將介紹一個可以生成廣義藏頭詩的系統，使用者可以彈性的決定自己所要藏的訊息以及其位子。與先前生成藏頭詩相關研究不同之處在於，以往的多是用硬性的規則來生成，本篇研究將使用 Seq2Seq 模型來生成藏頭詩。除了藏入訊息以外，使用者也可以指定韻腳以及句子長度，這是目前我們所知道第一篇可以做到以如此細微程度去控制模型生成句子的研究。

關鍵字： 自然語言生成, 藏頭詩生成, 自然語言處理, 機器學習, 創意寫作

doi: 10.6342/NTU201902806

Abstract

An acrostic is a form of writing that 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, this work adopts a neural-based sequence-to-sequence model to achieve this goal. Besides acrostic, users are also allowed to specify the rhyme and length of the output sequences. Based on our knowledge, this is the first neural-based natural language generation system that demonstrates the capability of performing micro-level control over output sentences.

Keywords: Natural language generation, Acrostic generation, Natural language processing, Machine learning, Creative writing

doi: 10.6342/NTU201902806

Contents

doi: 10.6342/NTU201902806

List of Figures

doi: 10.6342/NTU201902806

doi: 10.6342/NTU201902806

List of Tables

doi: 10.6342/NTU201902806

Chapter 1

Introduction

Acrostic is a form of writing aiming at hiding messages in text, often used in sarcasm or to deliver private information. Nowadays Seq2Seq models have become a popular choice for text generation, including generating text from table [5], summaries [6], short-text conversations [7], machine translation [1, 9] 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. However, 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 the acrostic (i.e. micro-controlling the position of a specific token) and rhyme. In this work, an NLG system that allows the users to micro-control the generation of a Seq2Seq model without any post-processing is presented. Besides specifying the tokens and their corresponding locations for acrostic, the 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 [10] can be trained to control the generation of the text. The demo system focuses on lyrics of Chinese and English, which can be regarded as a writing style in between articles and poetry. A general version of acrostic writing is considered in this work, which means the users can arbitrarily choose the position to place acrostic tokens.

doi: 10.6342/NTU201902806

Chapter 2

Related Works

In previous works, English acrostic have been generated by searching for paraphrases in WordNet' s synsets [8]. The work defines the generation of acrostic as a search problem that find synonyms and then rewrite the original text. Synonyms that contain needed characters replace the corresponding words in the context to generate acrostic. Figure 2.1 is an example of how to generate acrostic that contains the word "Turing". Given the original phrase "high influential", the method proposed by the authors finds synonym "revolutionized" that contains the "r" in "Turing". No published paper has mentioned the generation of Chinese acrostic.

Line	**Operator ϕ**	**Text → paraphrased text**
3	synonym	`highly influential` → `**r**evolutionized`
4	hyphenation	`giving` → `giv-**i**ng`
5	tautology	`computation` → `defi-**n**ite computation`
6	synonym	`considered` → `re-**g**arded`

Figure 2.1: Paraphrasing operations using "Turing" as example [8]

Micro-level controlling on attention-based Seq2Seq model have been found in previous work [11]. The author aimed to generate new lyrics that have same part of speech (POS) and rhyme to the given lyric input. The author found that with the design shown in Figure 2.2, the model is able to generate the next line of lyric that meets the $control$ $signals$ (i.e. $S_{i+1}.pos$ and $S_{i+1}.rhy$) with accuracy up to about 80%. The work shows that the Seq2Seq model trained with designed input is able to generate corresponding output

doi: 10.6342/NTU201902806

that meets the part of speech and rhyme given in the input.

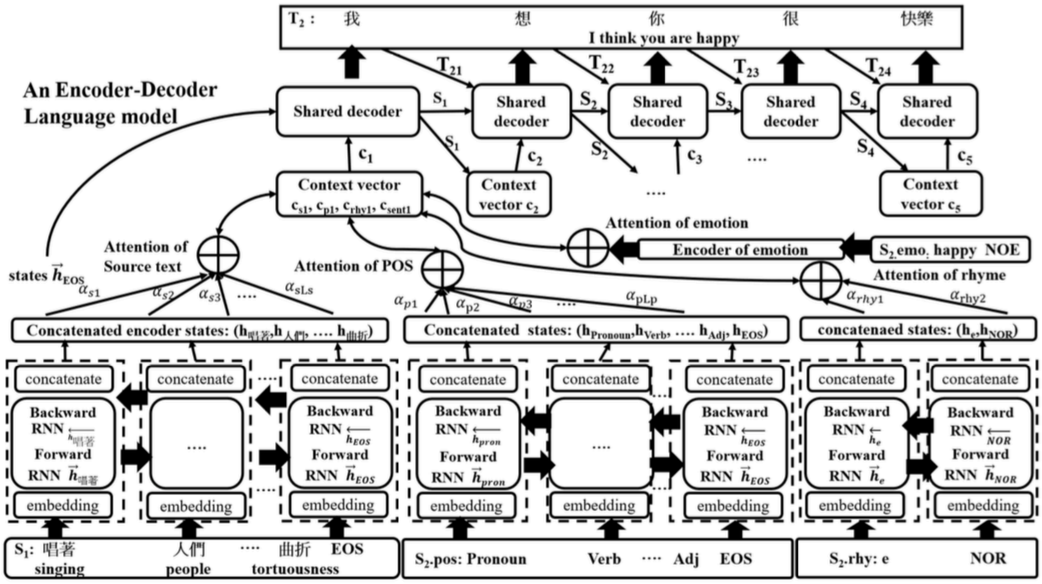

Figure 2.2: The multi-encoder model that have control signals as input [11]

doi: 10.6342/NTU201902806

Chapter 3

Model Description

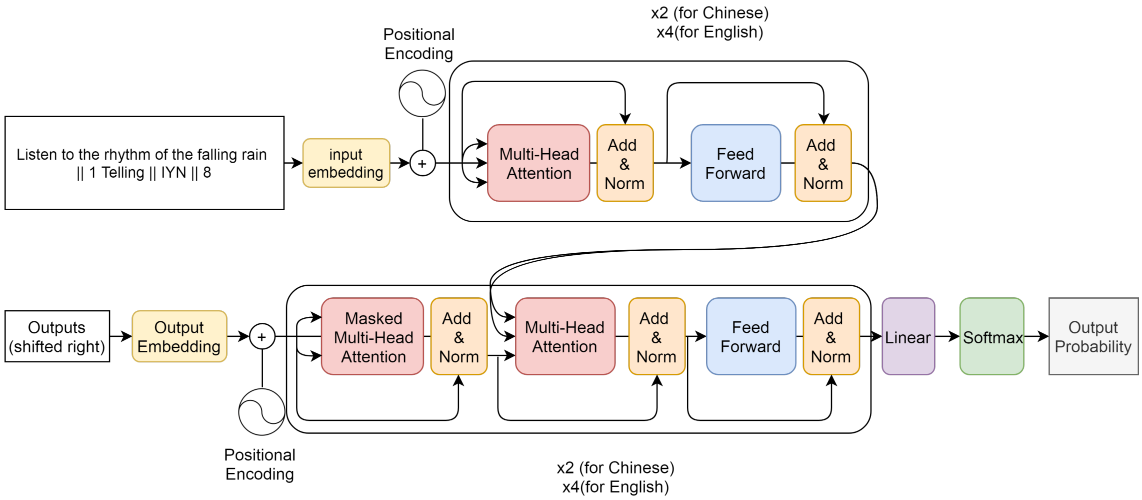

Figure 3.1: The structure of Transformer model.

Normally a neural-based Seq2Seq model is trained using input/output sequences as the training pairs [6, 3]. 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, a demo system is created to show that the users can indeed guide the outputs of a Seq2Seq model in a fine-grained manner. In the 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 can not only generate the next line

doi: 10.6342/NTU201902806

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 will use the 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 and from previous work [11], there is an interesting fact that by appending the $control\ signals$ in the end of the input sequences, then after seeing sufficient amount of such data, the Seq2Seq model can automatically discover the association between input signals and outputs. Once the associations are discovered, then users 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 $control\ signal$ indicate 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 (/ɪn/) and EYN corresponds to another (/en/). Note that here we use the

doi: 10.6342/NTU201902806

formal name of the rhyme (e.g. EYN) to improve the readability to humans, but 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 did we provide any prerequisite knowledge about what length, rhyme or position really stands for to the model.

The system is trained based on the Transformer model [10], though additional experiments show that other RNN-based Seq2Seq models such as the one based on GRU [2] or LSTM would also work. The model consists of an encoder and a decoder. Our encoder 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 [4] 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 3.1. Note that in the original paper [10], Transformer consists of six identical layers for both encoder and decoder. To save resource, we started training with fewer layers than the original paper and found that the model still performs well. Thus, we use fewer layers than the proposed Transformer model.

doi: 10.6342/NTU201902806

Chapter 4

System Structure

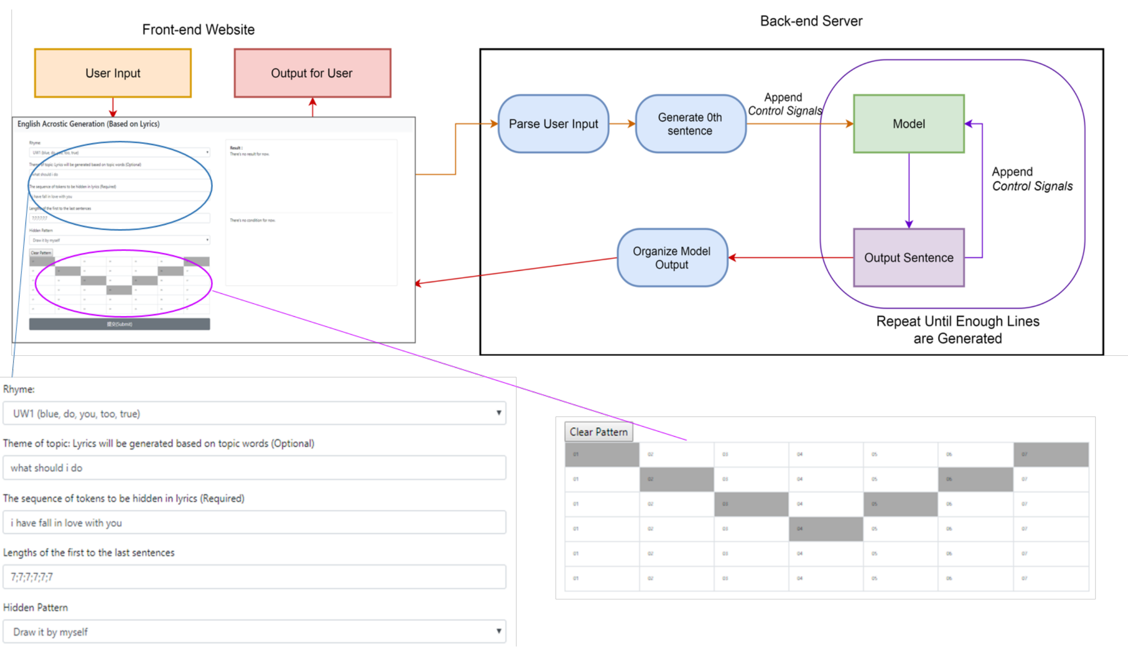

Figure 4.1: The structure of our acrostic generating system.

Figure 4.1 illustrates the interface and data flow of our acrostic lyric generating system. 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

doi: 10.6342/NTU201902806

for English Acrostic demonstration, 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 4.1, 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 use 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 the first sentence is appended to this $zeroth$ sequence to serve as the initial input to the Seq2Seq model for generating the first line of outputs. Next, the given condition of the second sentence is appended to the generated first line as the input to generate the second line. The same process is repeated until all lines are generated.

doi: 10.6342/NTU201902806

Chapter 5

Experiments

5.0.1 Data set

There are two versions of the acrostic generating system: one training on Chinese lyrics and one on English lyrics. The Chinese lyrics are crawled from Mojim lyrics site ($\mathrm{https://mojim.com/}$) and NetEase Cloud ($\mathrm{http://music.163.com/}$). To avoid rare characters, the vocabulary size is set to be the most frequent 50000 characters. The English lyrics are crawled from Lyrics Freak ($\mathrm{https://www.lyricsfreak.com/}$). The vocabulary size is set to be the most frequent 50000 words. For each lyric line, we first calculate its length, and 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 $control\ signal$, following 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 651339/1000000 training pairs we use to train our Chinese/English acrostic system. The validation set contains 30000 lines for both Chinese and English task.

doi: 10.6342/NTU201902806

5.0.2 Evaluation

The acrostic generating system has three controllable conditions on generating acrostic: the positions of designated tokens, the rhyme of each line and the length of each line.

To ensure the quality of generating acrostic lyrics, the accuracy of a character or a word on the designated position is very important. The following experiments start from training on word-based Chinese lyrics, character-based Chinese lyrics and then mixing the different $control\ signals$ together.

- Word-based training on Chinese lyrics of word position: The training pair is shown below. The $control\ signal$ here is "3 時代" which indicates the third word of the target line is "時代".

  s_1: 要 擁有 必先 懂 失去 怎 接受 || 3 時代

  $\rightarrow s_2$: 我 沿著 時代 去 戰鬥

  The accuracy of the word position tested on the validation set is 98.73% which indicates that the model learns the meaning of the $position\ control\ signal$.

- Word-based training on Chinese lyrics of character position: The training pair is shown below. The $control\ signal$ here is "4 時" which indicates the fourth character of the target line is "時".

  s_1: 要 擁有 必先 懂 失去 怎 接受 || 4 時

  $\rightarrow s_2$: 我 沿著 時代 去 戰鬥

  The accuracy of the character position tested on the validation set is 0.7716%. The model doesn't perform well on learning the character position while the sentence is segmented using words.

- Character-based training on Chinese lyrics of character position: The training pair is shown below. The $control\ signal$ here is "4 時" which indicates the fourth character of the target line is "時".

doi: 10.6342/NTU201902806

s_1: 要 擁 有 必 先 懂 失 去 怎 接 受 || 4 時

$\rightarrow s_2$: 我 沿 著 時 代 去 戰 鬥

The accuracy of the character position tested on the validation set is 91.62% which is better than using word-based training data.

- Character-based training on Chinese lyrics with multiple character positions: The training pair is shown below. The *control signal* here is "2 4 時 6 去" which indicates there are two designated character positions: the fourth character is "時" and the sixth character is "去".

  s_1: 要 擁 有 必 先 懂 失 去 怎 接 受 || 2 4 時 6 去

  $\rightarrow s_2$: 我 沿 著 時 代 去 戰 鬥

  The accuracy of the character position tested on the validation set is 99.32%.

- Character-based training on Chinese lyrics with multiple *control signals*: The training pair is shown below. The *control signal* here is "2 4 時 6 去 || ou || 8" which indicates there are two designated character positions: the fourth character is "時" and the sixth character is "去", the target line has the rhyme "ou" and there are eight characters in the target line.

  s_1: 要 擁 有 必 先 懂 失 去 怎 接 受 || 2 4 時 6 去 || ou || 8

  $\rightarrow s_2$: 我 沿 著 時 代 去 戰 鬥

  The accuracy is shown in Table 5.1.

- Word-based training on English lyrics with multiple *control signals*: The training pair is shown below. The *control signal* here is "1 7 me || IY1 || 7" which indicates there is one designated word position: the seventh word is "me", the target line has the rhyme "IY1" and there are seven words in the target line.

  s_1: look at her face it's a wonderful face || 1 7 me || IY1 || 7

  $\rightarrow s_2$: and it means something special to me

doi: 10.6342/NTU201902806

The accuracy is shown in Table 5.1.

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%

Table 5.1: The accuracy of each condition tested on 30000 lines and the perplexity of the original text and the text generated by our model.

As shown in Table 5.1, the model can almost perfectly satisfy the request from the users.

We also evaluate the quality of the 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 the generated lyrics becomes lower (42.33/42.34), which indicates the language model does learn a better way to represent the lyrics data.

Source	Perplexity(Chinese)	Perplexity(English)
Training data	54.56	53.2
Model generated data	42.33	42.34

Table 5.2: The perplexity of the original text and the text generated by our model.

In the experiments we found 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 observed that the model tends to generate sentences that use the same words appearing their previous sentences. This behavior might be learned from the repetition of lyrics lines.

doi: 10.6342/NTU201902806

Chapter 6

Demonstration samples

We provide our system outputs from different aspects.

The first example in Figure 6.1 shows that we can control the length of each line to produce a *triangle*-shape of lyrics.

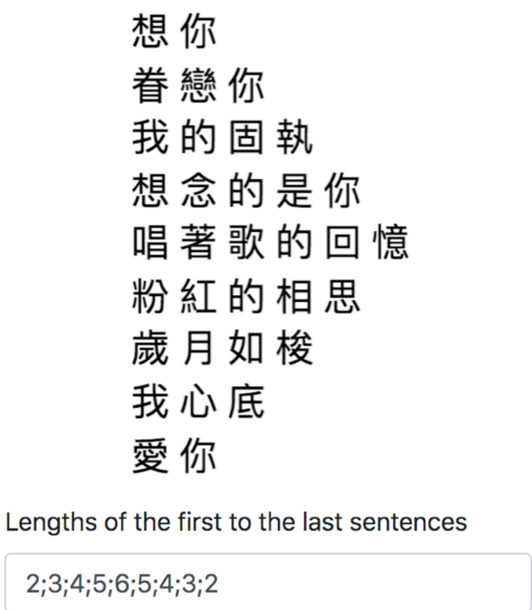

Figure 6.1: Length control of each sentence. The translation of generated lyrics: Missing you. Being sentimentally attached to you. I am so stubborn. Missing you. The memory of singing together. The pink lovesickness. Time flies like an arrow. I am still missing you from heart.

Second, we would like to demonstrate the results in generating acrostic. Some people use acrostic to hide message that has nothing to do with the major content of the text. We would show both English and Chinese examples generated by our system.

doi: 10.6342/NTU201902806

Figure 6.2 shows hiding a sentence in the first word of each sentences. The sentence that being concealed in the lyrics is *I don't like you* but the lyrics itself doesn't have that meaning.

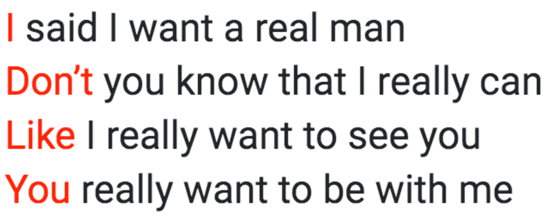

Figure 6.2: Message in English lyrics: *I don't like you*

Figure 6.3: Message in Chinese lyrics: 甚麼都可以藏 with rhyme *eng*. The translation of the generated lyrics: You don't even understand my dream. I don't know how to wait for things to happen, either. All because of the girl. All because of the dream. Anything can be turned into the wind. We can all have our own dreams.

Figure 6.3 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 says something about relationship and dream.

Third, we can also play with the visual shape of the designated words. Figure 6.4 shows an example 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 6.5

doi: 10.6342/NTU201902806

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

shows that we can hide the message using the shape of a heart.

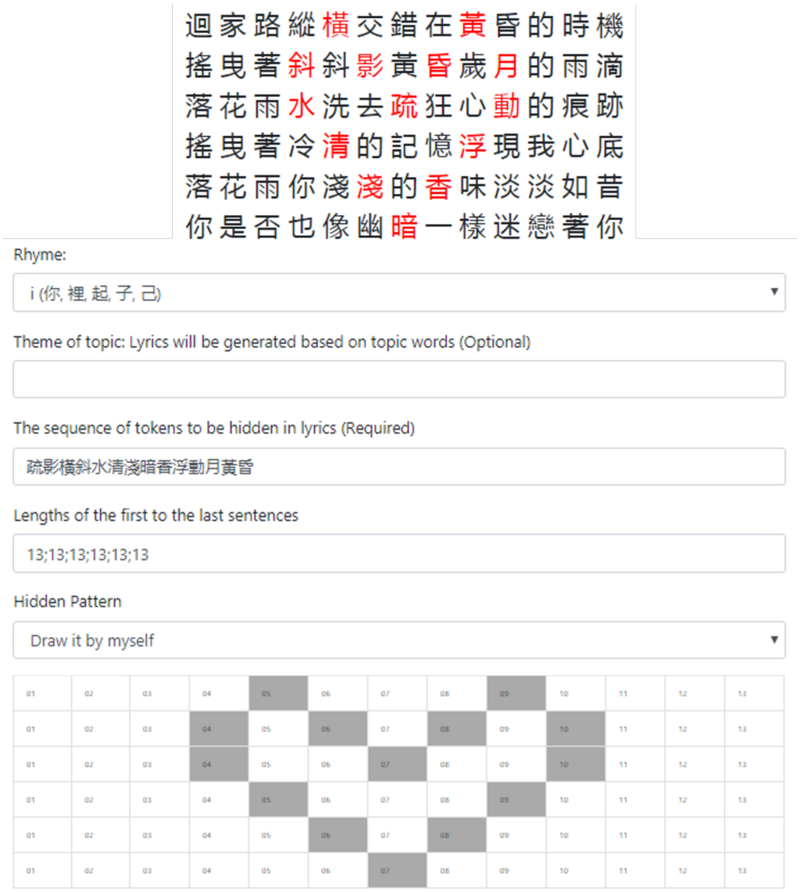

Figure 6.5: The designated characters form a heart. The sentence hidden in the lyrics is 疏影橫斜水清淺暗香浮動月黃昏 (The shadow reflects on the water and the fragrance drifts under the moon with the color of dusk) with rhyme i. The translation of the generated lyrics: The roads to home crisscross in the dusk. The rain drops and the shadow swing with the past times. Falling flowers and rain wash away my feelings for you. The lonely memory arises in my heart. With the falling flowers and rain, your light fragrance is as usual. Are you obsessed with yourself just as the darkness is?

doi: 10.6342/NTU201902806

Chapter 7

Conclusions

We show that by appending additional information to 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 can not only generate meaningful content but follow the constraints with very high accuracy. We believe that this finding can further lead to other useful applications in natural language generation.

doi: 10.6342/NTU201902806

Chapter 8

Future Work

Acrostic generation is just one possible application of control signals. There are still other interesting applications and different control signals for people to explore. Apart from application, the learning mechanism of the model is also an important researching topic. How does the model learn the meaning of the control signals? Where does the model store the information? If we can know how the model learns, it may be easier for us to control the generation of natural language.

doi: 10.6342/NTU201902806

Bibliography

[1] D. Bahdanau, K. Cho, and Y. Bengio. 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, 2015.

[2] K. Cho, B. van Merriënboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. 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, Oct. 2014. Association for Computational Linguistics.

[3] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. 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, 2014.

[4] K. He, X. Zhang, S. Ren, and J. Sun. 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, 2016.

[5] T. Liu, K. Wang, L. Sha, B. Chang, and Z. Sui. 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

doi: 10.6342/NTU201902806

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, 2018.

[6] R. Nallapati, B. Zhou, C. N. dos Santos, Ç. Gülçehre, and B. Xiang. 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, 2016.

[7] L. Shang, Z. Lu, and H. Li. 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, 2015.

[8] B. Stein, M. Hagen, and C. Bräutigam. 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, 2014.

[9] I. Sutskever, O. Vinyals, and Q. V. Le. 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, 2014.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. 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, 2017.

[11] C.-C. Wu. An attention based neural network model for unsupervised lyrics rewriting. 2018.

doi: 10.6342/NTU201902806