Stylistic Chinese Poetry Generation via Unsupervised Style Disentanglement

Cheng Yang[1,2,3], **Maosong Sun**[1,2,4*], **Xiaoyuan Yi**[1,2,3], **Wenhao Li**[1,2,3]
[1]Department of Computer Science and Technology, Tsinghua University
[2]Institute for Artificial Intelligence, Tsinghua University
[3]State Key Lab on Intelligent Technology and Systems, Tsinghua University
[4]Jiangsu Collaborative Innovation Center for Language Ability, Jiangsu Normal University
{cheng-ya14,yi-xy16,liwh16}@mails.tsinghua.edu.cn
sms@tsinghua.edu.cn

Abstract

The ability to write diverse poems in different styles under the same poetic imagery is an important characteristic of human poetry writing. Most previous works on automatic Chinese poetry generation focused on improving the coherency among lines. Some work explored style transfer but suffered from expensive expert labeling of poem styles. In this paper, we target on stylistic poetry generation in a fully unsupervised manner for the first time. We propose a novel model which requires no supervised style labeling by incorporating mutual information, a concept in information theory, into modeling. Experimental results show that our model is able to generate stylistic poems without losing fluency and coherency.

1 Introduction

Classical Chinese poetries are great heritages of the history of Chinese culture. One of the most popular genres of classical Chinese poems, *i.e. quatrains*, contains four lines with five or seven characters each and additional rhythm and tune restrictions. During 1,000 years history of quatrains, various styles, *e.g.* pastoral, descriptive and romantic, have been developed to express different feelings of poets. In human poetry writing, poets are able to write completely different poems in diverse styles even given the same keyword or first sentence. For example, as shown in Fig. 1, when a poet mentioned " 月" (the moon), she/he may write about the Great Wall in the battlefield style or the sleepless feeling in the romantic style. Such ability to write stylistic poems under the same poetic imagery is an important characteristic of human poetries.

Automatic poetry generation is one of the first attempts towards computer writing. Chinese quatrain generation has also attracted much attention in recent years. Early works inspired by statistical machine translation explored rule-based and template-based methods (He et al., 2012; Yan et al., 2013), while recent works (Zhang and Lapata, 2014; Wang et al., 2016; Yan, 2016; Zhang et al., 2017; Yang et al., 2017; Yi et al., 2017) employed neural network based sequence-to-sequence approaches which have shown their effectiveness in neural machine translation for poem generation. Most works target on improving the coherency among all lines and the conformity between the theme and subsequent lines by planning (Wang et al., 2016), polishing schema (Yan, 2016), poem block (Yi et al., 2017) and conditional variational autoencoder (Yang et al., 2017). Different from these previous works, we aim to learn the ability of diverse stylistic poetry generation which can generate multiple outputs (poems) in various styles under the same input (keywords or the first sentence). This ability enables a poetry generation system to be closer to a real poet and allows the model to generate more expressive and creative poems.

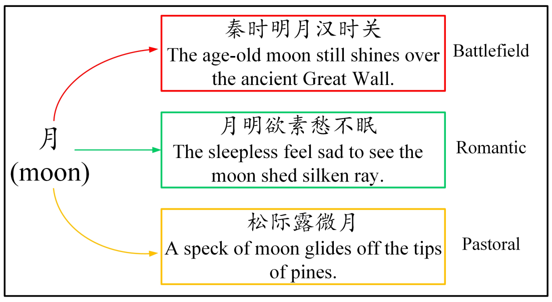

月
(moon)
秦时明月汉时关
The age-old moon still shines over the ancient Great Wall.
Battlefield
月明欲素愁不眠
The sleepless feel sad to see the moon shed silken ray.
Romantic
松际露微月
A speck of moon glides off the tips of pines.
Pastoral

Figure 1: An example of poems in diverse styles under the same keyword.

However, there is no explicit label about what style or category a poem or a sentence is for thousands of poems in the database. Therefore, traditional supervised sequence-to-sequence mod-

[*] corresponding author: sms@tsinghua.edu.cn

els (Zhang et al., 2017) are not capable to generate stylistic poems without expert labeling. To the best of our knowledge, we are the first effort at stylistic poetry generation in a fully unsupervised manner.

In this paper, we propose a novel poetry generation model which can disentangle the poems in different styles and generate style-specific outputs conditioned on the manually selected style input. We employ sequence-to-sequence model with attention mechanism (Bahdanau et al., 2014) as our basis and maximize the mutual information which measures the dependency between two random variables in information theory to strengthen the relationship between manually selected style inputs and generated style-specific outputs. Experimental results show that our model is able to generate poems in various styles without losing fluency and coherency.

To summarize, our effort provides the following three contributions:

• To the best of our knowledge, we are the first effort at stylistic poetry generation which is an important characteristic of human poetries in a fully unsupervised manner.

• We innovatively incorporate mutual information, a measurement in information theory, for unsupervised style disentanglement and style-specific generation.

• Experimental results show that our model is able to generate diverse poems in various styles without losing fluency and coherency.

2 Related Works and Motivations

Poetry generation is a classic task in computer writing (Gervás, 2001; Levy, 2001; Netzer et al., 2009; Oliveira, 2012). Chinese poetry generation has also attracted much attention during the last decade. Early works are mostly rule-based and template-based (Wu et al., 2009). (He et al., 2012) employed statistical machine translation and (Yan et al., 2013) adopted automatic summarization techniques for classical poem generation.

As neural network based approaches have been successfully applied in various applications such as machine translation (Bahdanau et al., 2014), people came up with the idea to apply sequence-to-sequence models for poem generation. (Zhang and Lapata, 2014) employed the Recurrent Neural Network (RNN) as their basis and further considered the global context using Convolutional Neural Network (CNN). They also incorporated other hand-crafted features into the model to improve the coherency. (Yan, 2016) proposed an iterative polishing schema based on two RNNs, which can refine the generated poems for several times. (Wang et al., 2016) generated poetries in a two-stage process: planning the keywords of each line first and then generating each line sequentially. (Yi et al., 2017) proposed poem blocks to learn semantic meaning within a single line and semantic relevance among lines in a poem. (Yang et al., 2017) employed a conditional variational autoencoder with augmented word2vec architecture to enhance the conformity between the theme and generated poems. All previous works above target on improving the coherency and conformity of poetry generation while a recent work (Zhang et al., 2017) was proposed to improve the novelty of generated poems using external memories. Their model can also transfer the style of a poem into three predefined topics. However, they need to manually label many poems in these three predefined topics to learn the patterns.

Formally, a traditional sequence-to-sequence model actually learns a conditional probability distribution $\Pr(s_{output}|s_{input})$ where s_{output} is the generated poems and s_{input} is the input keyword or the first sentence. Different from machine translation tasks where the output sentence s_{output} is rather certain and compact given input sentence s_{input}, the conditional probability distribution $\Pr(s_{output}|s_{input})$ has strong uncertainty in literary creation scenarios such as poem generation. To support this point, we further train a topic model based on LDA (Blei et al., 2003) by treating each poem as a document. We train a 10-topic and a 20-topic LDA model and then reanalyze the largest topic component of each sentence in a poem. Surprisingly, we find that only 20% and 10% consecutive sentences in a poem have the same largest topic components respectively though we assume that each poem is generated by the same topic when training the LDA model. The fact indicates that even given the former sentence, the style of the latter one could still be diverse and flexible. Intuitively, poets will choose a style or topic in their minds and then write the next sentence based on the style they choose in poetry creation. Therefore, we propose to learn a stylistic poetry generation model which can disentangle the poems in different styles and generate style-specific outputs as real human poets could do. Note that no explicit style labels are given to the training data and thus previous supervised algorithms cannot be adapted for the purpose easily.

Recently, a few works have been proposed for disentangled representation learning. InfoGAN (Chen et al., 2016) was proposed for generating continuous image data conditioned on image labels and disentangled text generation (Hu et al., 2017) focused on controllable generation in the semi-supervised setting. Though inspired by them, the motivation and proposed models of our work differ from these methods by a large margin. To the best of our knowledge, we are the first effort at stylistic poetry generation in a fully unsupervised manner.

3 Method

We hope our generation model can generate multiple outputs in various styles. Formally, our model takes two arguments as input: input sentence s_{input} and style id $k \in 1, 2 \dots K$ where K is the total number of different styles. Then we can enumerate each style id k and generate style-specific output sentence s^k_{output} respectively.

In this section, we will start by introducing the background knowledge about mutual information and sequence-to-sequence model with attention mechanism. Then we propose our framework of decoder model for style disentanglement by taking the mutual information as an additional regularization term. Finally, we will present the details of our implementations of each component in the framework.

3.1 Mutual Information

Inspired by previous works on image generation (Chen et al., 2016) and semi-supervised generation (Hu et al., 2017), we propose to incorporate the concept of mutual information in information theory for style disentanglement. Given two random variables X and Y, the mutual information $I(X, Y)$ measures “the amount of information” obtained about one random variable given another one [1]. Mutual information can also be interpreted as a measurement about how similar the joint probability distribution $p(X, Y)$ is to the product of marginal distributions $p(X)p(Y)$. The definition of mutual information is

$$I(X,Y) = \int_Y \int_X p(X,Y) \log \frac{p(X,Y)}{p(X)p(Y)} dXdY. \quad (1)$$

3.2 Sequence-to-Sequence Model with Attention Mechanism

We employ a widely used Encoder-Decoder framework (Sutskever et al., 2014) which was firstly proposed in machine translation as our basis. Suppose sentence $X = (x_1 x_2 \dots x_T)$ and $Y = (y_1 y_2 \dots y_T)$ are the input sentence (source sentence) and output sentence (target sentence) respectively where x_i, y_i for $i = 1, 2 \dots T$ are characters and T is the total number of characters in the sentence. We denote the character vocabulary as V.

Specifically, we use bidirectional LSTM (biLSTM) (Hochreiter and Schmidhuber, 1997; Schuster and Paliwal, 1997) model as the encoder to project the input sentence X into the vector space. Formally, the hidden state of LSTM are computed by

$$\overrightarrow{h_i} = LSTM_{forward}(\overrightarrow{h_{i-1}}, e(x_i)), \quad (2)$$

$$\overleftarrow{h_i} = LSTM_{backward}(\overleftarrow{h_{i-1}}, e(x_{T-i+1})), \quad (3)$$

for $i = 1, 2 \dots T$ where $\overrightarrow{h_i}$ and $\overleftarrow{h_i}$ are the i-th hidden state of forward and backward LSTM respectively, $e(x_i) \in \mathcal{R}^d$ is the character embedding of character x_i and d is the dimension of character embeddings. Then we concatenate corresponding hidden states of forward and backward LSTM $h_i = [\overrightarrow{h_i}, \overleftarrow{h_{T-i+1}}]$ as the i-th hidden state of biLSTM. In specific, we use the last hidden state h_T as the embedding vector and feed it to the decoder.

The decoder module contains an LSTM decoder with attention mechanism (Bahdanau et al., 2014) which computes a context vector as a weighted sum of all encoder hidden states to represent the most relevant information at each stage. The character probability distribution when decoding the i-th character can be expressed as

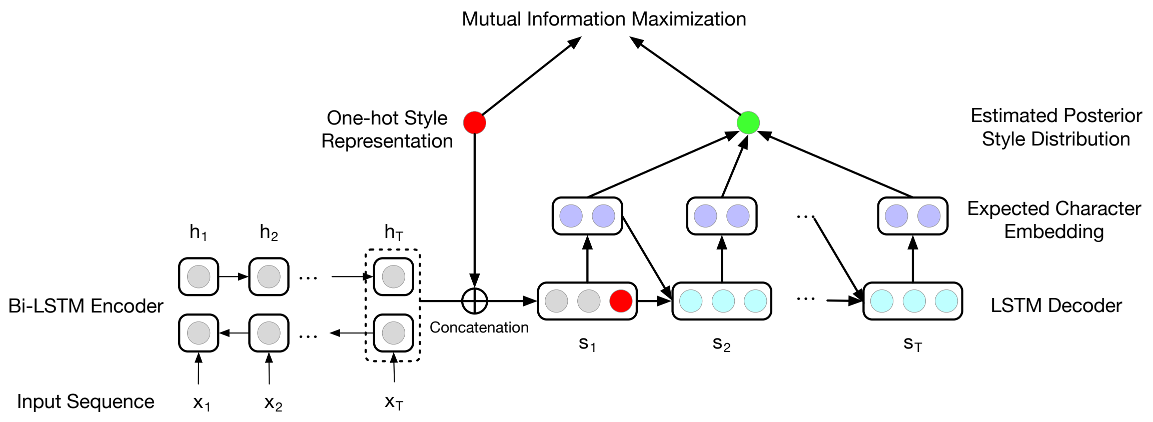
$$p(y_i | y_1 y_2 \dots y_{i-1}, X) = g(y_i | s_i), \quad (4)$$

where $g(\cdot)$ is a linear projection function with softmax regularization, s_i is the i-th hidden state in the decoder LSTM:

$$s_i = LSTM_{decoder}(s_{i-1}, [e(y_{i-1}), a_i]), \quad (5)$$

for $i = 2, \dots T$ and $s_1 = h_T$, where [ ] indicates concatenation operation, $e(y_{i-1})$ is character embedding of y_{i-1} and a_i is the context vector

[1] https://en.wikipedia.org/wiki/Mutual_information



Mutual Information Maximization
One-hot Style Representation
Estimated Posterior Style Distribution
Expected Character Embedding
Bi-LSTM Encoder
LSTM Decoder
Concatenation
Input Sequence
h1
h2
hT
x1
x2
xT
s1
s2
sT

Figure 2: An overview of style disentanglement by mutual information maximization.

learned by attention mechanism (Bahdanau et al., 2014)

$$a_i = attention(s_{i-1}, h_{1:T}). \tag{6}$$

3.3 Decoder Model with Style Disentanglement

To accept two arguments, *i.e.*, input sentence X and style id k, we can directly concatenate the one-hot representation of style id $one\_hot(k)$ and the embedding vector h_T obtained by bi-LSTM encoder and then feed the concatenated vector $[one\_hot(k), h_T]$ instead of h_T into the decoder model without changing anything else.

However, there is no theoretical guarantee that the output sentence generated by the decoder is strongly correlated to the style id input $one\_hot(k)$. In other words, when the input style id is changed, the output sentence would probably be the same because no supervised loss is given to force the output sentences to follow the one-hot style representation. Therefore, we naturally come up with the idea to add a regularization term to force a strong dependency relationship between the input style id and generated sentence.

Without loss of generality, we assume that the input style id is a uniformly distributed random variable Sty and $\Pr(Sty = k) = \frac{1}{K}$ for $k = 1, 2 \ldots K$ where K is the total number of styles. Recall that mutual information quantifies the mutual dependency between two random variables. Hence we propose to maximize the mutual information between the style distribution $\Pr(Sty)$ and the generated sentence distribution $\Pr(Y; X)$ given input sentence X to strengthen the dependency between them as shown in Fig. 2. The mutual information is computed as

$$\begin{aligned}
&I(\Pr(Sty), \Pr(Y; X)) \\
=&\sum_{k=1}^{K}\Pr(Sty = k)\int_{Y|k;X}\log\frac{\Pr(Y, Sty = k; X)}{\Pr(Sty = k)\Pr(Y; X)}dY \\
=&\sum_{k=1}^{K}\Pr(Sty = k)\int_{Y|k;X}\log\frac{\Pr(Y, Sty = k; X)}{\Pr(Y; X)}dY \\
&-\sum_{k=1}^{K}\Pr(Sty = k)\log\Pr(Sty = k) \\
=&\sum_{k=1}^{K}\Pr(Sty = k)\int_{Y|k;X}\log\Pr(Sty = k|Y)dY + \log K \\
=&\int_{Y;X}\sum_{k=1}^{K}\Pr(Sty = k|Y)\log P(Sty = k|Y)dY + \log K.
\end{aligned} \tag{7}$$

Note that the input sequence X and style Sty are both input arguments and thus independent with each other. Therefore, posterior distribution $\Pr(Sty = k|Y; X) = \Pr(Sty = k|Y)$.

But the posterior probability distribution $\Pr(Sty = k|Y)$ is actually unknown, we cannot compute the integration directly. Fortunately, with the help of variational inference maximization (Barber and Agakov, 2003), we can train a parameterized function $Q(Sty = k|Y)$ which estimates the posterior distribution and maximize a lower bound of mutual information in Eq. 7 instead:

$$\begin{aligned}
&I(\Pr(Sty), \Pr(Y; X)) - \log K \\
&= \int_{Y;X}\sum_{k=1}^{K}\Pr(Sty = k|Y)\log\Pr(Sty = k|Y)dY \\
&= \int_{Y;X}\sum_{k=1}^{K}\Pr(Sty = k|Y)\log Q(Sty = k|Y)dY \\
&\quad + \int_{Y;X}\sum_{k=1}^{K}\Pr(Sty = k|Y)\log\frac{\Pr(Sty = k|Y)}{Q(Sty = k|Y)}dY
\end{aligned}$$

$$
\begin{aligned}
&= \int_{Y;X} \sum_{k=1}^{K} \Pr(Sty = k|Y) \log Q(Sty = k|Y) dY \\
&\quad + \int_{Y;X} KL(\Pr(Sty|Y), Q(Sty|Y)) dY \\
&\geq \int_{Y;X} \sum_{k=1}^{K} \Pr(Sty = k|Y) \log Q(Sty = k|Y) dY \\
&= \sum_{k=1}^{K} \Pr(Sty = k) \int_{Y|k;X} \log Q(Sty = k|Y) dY.
\end{aligned} \tag{8}
$$

Here $KL(\Pr(\cdot)||Q(\cdot))$ indicates the KL-divergence distance between probability distribution $\Pr(\cdot)$ and $Q(\cdot)$. The inequality comes from the fact that the KL-divergence is always no less than zero and is tight when $\Pr(\cdot) = Q(\cdot)$.

We use the lower bound parameterized by function Q in Eq. 8 as an additional maximization term. Intuitively, the lower bound is maximized if we can perfectly infer the style id from generated style-specific outputs by inference function Q. It also indicates that generated outputs will heavily depend on the input style id and outputs generated by different styles are distinguishable, which follows our motivation of stylistic poetry generation. Now we will introduce how to design the posterior distribution estimation function Q and compute the integration in the lower bound.

3.4 Posterior Distribution Estimation

Given an output sequence Y, the function Q estimates the probability distribution of the style of sequence Y and therefore disentangles different styles. In this paper, we employ neural network to parametrize the posterior distribution estimation function Q. Specifically, we first compute the average character embeddings of sequence Y and then use a linear projection with softmax normalizer to get the style distribution. Formally, $Q(Sty|Y)$ is computed as

$$
Q(Sty|Y) = softmax(W \cdot \frac{1}{T} \sum_{i=1}^{T} e(y_i)), \tag{9}
$$

where $W \in \mathcal{R}^{K \times d}$ is the linear projection matrix.

Then the last thing to do is to compute the integration over $Y|k;X$. However, the search space of sequence Y is exponential to the size of vocabulary. Hence it's impossible to enumerate all possible sequence Y for computing the integration. Also, it is not differentiable if we sample sequence Y according to generation probability for approximation. Therefore, we propose to use expected character embedding to approximate the integration.

3.5 Expected Character Embedding

Inspired by previous works (Kočiský et al., 2016), we use expected character embedding to approximate the probabilistic space of output sequences: we only generate an expected embedding sequence and suppose $Y|k;X$ has one hundred percent probability generating this one. Formally, Eq. 4 gives a probability distribution of generating the i-th character given previous ones. Then the "expected" generated character embedding is

$$
expect(i; k, X) = \sum_{c \in V} g(c|s_i) e(c), \tag{10}
$$

where $expect(i; k, X) \in \mathcal{R}^d$ represents the expected character embedding at i-th output given style id k and input sequence X and $c \in V$ enumerates all characters in the vocabulary.

Then $expect(i; k, X)$ is fed into the LSTM in decoder to update the hidden state for generating next expected character embedding:

$$
s_{i+1} = LSTM_{decoder}(s_i, [expect(i; k, X), a_{i+1}]). \tag{11}
$$

Finally, we use the expected embeddings $expect(i; k, X)$ for $i = 1, 2 \ldots T$ as an approximation of the whole probability space of $Y|k;X$. The lower bound in Eq. 8 can be rewritten as

$$
\mathcal{L}_{reg} = \frac{1}{K} \sum_{k=1}^{K} \log\{softmax(W \cdot \frac{1}{T} \sum_{i=1}^{T} expect(i; k, X))[k]\}, \tag{12}
$$

where $x[j]$ represents the j-th dimension of vector x.

We add the lower bound $\mathcal{L}_{reg}$ to the overall training objective as a regularization term. The computing process is shown in Fig. 2. Then for each training pair (X, Y), we aim to maximize

$$
Train(X, Y) = \sum_{i=1}^{T} \log p(y_i|y_1 y_2 \ldots y_{i-1}, X) + \lambda \mathcal{L}_{reg}, \tag{13}
$$

where $p(y_i|y_1 y_2 \ldots y_{i-1}, X)$ is style irrelevant generation likelihood and computed by setting one-hot style representation to an all-zero vector, and λ is a harmonic hyperparameter to balance the log-likelihood of generating sequence Y and the lower bound of mutual information. The first term ensures that the decoder can generate fluent and coherent outputs and the second term guarantees the style-specific output has a strong dependency on the one-hot style representation input.

Note that unlike machine translation task where the trained model is desired to generate exactly the same as the target, poetry generation encourages novelty as an important requirement. In Eq. 13, we can also enumerate the style id representation as a one-hot vector instead of an all-zero one. However, this will force the generation of all styles to be close to the training target and discourage the diversity and novelty.

Moreover, our model is not task-specific: the regularization term $\mathcal{L}_{reg}$ can be added to other models conveniently for diverse or stylistic generations. A Chinese quatrain has at most 4*7=28 characters and clear rhyme requirements. As the first attempt on unsupervised style disentanglement, we find Chinese poetry generation is an ideal choice for evaluation because we can better focus on stylistic generation rather than dealing with genre requirements. We will explore the applicability of our model on other languages and tasks for future work.

4 Experiments

Following the experimental settings in previous works (Zhang et al., 2017), we conduct human judgment to evaluate the performance of our model and state-of-the-art baseline methods. We will first introduce the dataset, our model details, baseline methods and evaluation settings. Then we will present the experimental results and give further analysis about the evaluation. Finally, we will present some example generations for case study.

4.1 Dataset and Model Details

We collect 168,000 poems (half *wuyan*: five characters per line, half *qiyan*: seven characters per line) over 1,000 years history of classical Chinese poetries as our corpus. We randomly select 80% of the corpus as training set, 10% as validation set and leave the rest for test. We extract all consecutive sentences in a poem as training pairs (X, Y) and feed them into our model.

For hyperparameter settings of our model, *i.e.* stylistic poetry generation (SPG), we set the dimensions of character embeddings and encoder hidden states as $d = 512$. The total number of styles is set to $K = 10$. Hence the dimension of decoder hidden states is $512 + 512 + 10 = 1034$. We pack the training pairs to mini-batches and the size of mini-batches is set to 50. The harmonic hyperparameter is set to $\lambda = 0$ for the first $50,000$ mini-batches as pretraining and $\lambda = 1.0$ for subsequent batches. We use Adam optimizer (Kingma and Ba, 2014) for stochastic gradient descent. We also employ dropout (Srivastava et al., 2014) strategy with dropout rate 0.2 to avoid overfitting problem. We terminate the training process when the optimization loss on validation set is stable, *i.e.* $300,000$ mini-batches in total.

4.2 Baselines

We consider the following state-of-the-art poetry generation models for comparison:

- **Seq2seq** (Bahdanau et al., 2014) is the sequence-to-sequence model with attention mechanism. Note that seq2seq is the basis of our SPG model. We can better analyze the improvement by style disentanglement from the comparison with seq2seq.
- **Polish** (Yan, 2016) proposed an iterative schema to polish and refine the generated sentences for several times instead of a one-pass generation.
- **Memory** (Zhang et al., 2017) incorporated external memory into poem generation for better novelty. The memory can be seen as a regularization to constrain the behavior of the neural model.

Rule-based and template-based methods are not considered as our baselines as they have been already thoroughly compared in (He et al., 2012; Yan, 2016).

4.3 Evaluation Settings

Following the settings in (Zhang et al., 2017), we employ human judgment for evaluation. To better compare the ability to generate fluent and coherent poems, we fix the first sentence as input and let all models generate three subsequent sentences. The first sentences are randomly chosen from the poems in the test set. Therefore we also consider the original poems written by real poets for comparison.

Note that our SPG model need a manually specified style id as input. For fully automatic poem generation, we use the posterior style estimation function $Q(\cdot)$ to infer the style of the first sentence and then generate next three sentences sequentially using the same style.

As previous works (Manurung, 2004; Yi et al., 2017) did, we design four criteria for human judgment: **Fluency** (are the generated poems fluent and well-formed?), **Coherence** (is the topic of the whole quatrain consistent?), **Meaningfulness**

Group	Methods	Fluency		Coherence		Meaningfulness		Poeticness	
		wuyan	qiyan	wuyan	qiyan	wuyan	qiyan	wuyan	qiyan
Group 1	seq2seq	2.75	2.48	2.60	2.33	2.38	2.15	2.40	2.35
	SPG	**3.43**	**3.23**	**3.13**	**3.05**	**2.83**	**3.10**	**3.10**	**3.10**
Group 2	memory	2.50	2.60	2.30	2.38	2.18	2.25	2.05	2.40
	polish	2.53	2.90	2.28	2.55	2.15	2.55	2.20	2.50
	SPG	**3.38**	**3.53**	**3.38**	**3.30**	**3.13**	**3.25**	**3.20**	**3.20**
	human poet	3.85	3.68	4.05	3.83	4.00	3.83	3.53	3.75

Table 1: Human judgment results. We bold the best performing model in each group.

(does the poem convey some certain messages?) and **Poeticness** (does the poem have some poetic features?). Each criterion needs to be scored on a 5-point scale ranging from 1 to 5.

For each model, we generate 20 *wuyan* and 20 *qiyan* quatrains given the first sentence. We invite 10 experts who major in Chinese literature or are members of a poetry association to evaluate these quatrains. We divide the baseline methods into two groups: In the first group, we compare SPG with seq2seq (Bahdanau et al., 2014) to present the advantages of style disentanglement; In the second group, we compare SPG with state-of-the-art poetry generation methods, memory (Zhang et al., 2017) and polish (Yan, 2016), and the original poems written by poets to demonstrate the effectiveness of our algorithm. Note that the scores of human judgment are relative, not absolute. Thus we compare with seq2seq separately to avoid mutual interference.

4.4 Experimental Results

We report the average scores of expert judgments in Table 1. From the experimental results, we have the following observations:

Firstly, the experimental results in group 1 demonstrate that SPG outperforms seq2seq, the basis of our model, by learning a style-disentangled generation model. The only difference between seq2seq and our model is that we append a one-hot style id to the encoder state and add mutual information regularization in Eq. 12 to the loss function. Note that our model is fully unsupervised: the one-hot style id conveys no meaningful message unless the mutual information regularization is considered. In other words, the one-hot style id and mutual information regularization cannot be torn apart. Therefore, the improvements over seq2seq all come from the part of style disentanglement modeling because seq2seq and our model share all the other components. The comparisons in group 1 of Table 1 are sufficient to show the effectiveness of style modeling.

Secondly, the experimental results in group 2 show that SPG consistently outperforms two state-of-the-art poem generation methods. SPG is able to generate poetries in diverse styles without losing fluency and coherency. Our advantages are two-fold: On one hand, SPG better fits the diversity characteristic of human poetry writing. Intuitively, seq2seq learns a generation model that mixes poems in various styles and is more likely to generate meaningless common sentences. By disentangling poems in different styles, our model can learn a more compact generation model for each style and write more fluent and meaningful poems. On the other hand, by fixing the style when generating three lines in a quatrain, SPG is able to generate more coherent poems.

One can also imagine that the generation probability distribution actually consists of several peaks where each peak identifies a cluster of poems in similar styles. A traditional model mixes all the stuff together and learns a one-peak generation model which is more likely to generate meaningless common sentences. In contrast, our model disentangles different clusters (styles) and learns a more accurate and compact generation model for each cluster (style). Hence our model can generate poems with higher quality and beat the baselines in terms of human evaluation. This observation demonstrates the effectiveness and robustness of our model.

Finally, there is still a large margin between SPG and human poets. In this paper, we focus on poem generation in diverse styles. Other poetry characteristics are also important for the quality of generations. We will consider applying our style-disentangle regularization on other poem models for better performance in the future.

Style id	Keywords
1	loneliness, melancholy
2	the portrait of landscape
3	sorrow during roaming
4	hermit, rural scenes
5	grand scenery, regrets about old events
6	sorrow during drinking
7	emotions towards life experience
8	the portrait of hazy sceneries
9	reminiscence, homesickness
10	sadness about seasons

Table 2: Representative keywords for poems generated by each learned style.

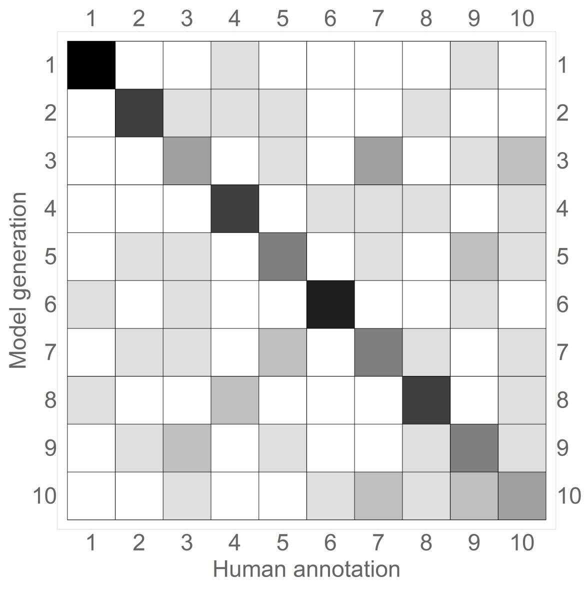


Figure 3: Experimental results on style recognition. Each row represents the human annotation of corresponding style generations. The diagonal blocks are correct classifications. Darker block indicates higher probability.

4.5 Interpretability of Learned Styles

In this subsection, we conduct experiments to understand the semantic meanings of each learned style. To this end, we first generate 50 poems in each style and then manually select two or three keywords to sketch the overall feelings of every style. The keywords of all 10 styles are listed in Table 2. Note that the learned styles may not strictly align the traditional writing styles recognized by human such as romantic or pastoral because our model is fully unsupervised.

Then we generate another 50 poems (5 poems per style) and ask the experts to classify these poems into the 10 styles according to the relationship between generated poems and style keywords in Table 2. The human annotation results are shown in Fig. 3.

We can see that many learned styles can be successfully recognized by human with a higher probability, e.g. the first style can be correctly classified by 80% probability. This observation indicates that the learned styles of SPG are meaningful and recognizable through only two or three keywords. Also, the generated poems are diverse otherwise they cannot be differentiated and correctly classified.

4.6 Case Study

We present three poems generated by SPG with the same first sentence for case study. We only list the results of three most representative styles in Fig. 4 and put other generation examples in supplementary materials (in Chinese). We can see that the poems generated by different style inputs differ a lot from each other and follow the style keywords. To conclude, SPG can generate fluent and coherent poetries in diverse styles.

5 Conclusion

In this paper, we propose a stylistic poetry generation (SPG) model to learn the ability to write poems in different styles under the same poetic imagery. To the best of our knowledge, we are the first effort at stylistic poetry generation in a fully unsupervised manner. Therefore, our model requires no expensive expert style annotation for thousands of poems in the database. We innovatively employ mutual information, a concept in information theory, to develop our model. Experimental results show that SPG is able to generate fluent and coherent poetries in diverse styles without losing fluency and coherency. The learned styles are meaningful and recognizable given only two or three keywords. Our algorithm has been incorporated into Jiuge[2] poetry generation system.

For future works, we will consider adopting the mutual information regularization for other text generation tasks which encourage stylistic generation or diversity to improve their performances. Another intriguing direction is to refine our model for more task-specific scenarios. Besides, our model simplifies the prior of style distribution as a uniform distribution. This assumption could be further improved by an iteratively updating approach.

[2]https://jiuge.thunlp.cn

浊酒一杯聊酩酊，
After a cup of unstrained wine,
I have been a little drunk
白云千里断鸿濛。
I saw the cloud split the sky apart.
马蹄踏破青山路，
On horseback, I pass through every road across the mountain,
惆怅斜阳落日红。
but can only watch the red sun falling down with sorrow.

(a) Style 1: “loneliness, melancholy”

浊酒一杯聊酩酊，
After a cup of unstrained wine,
I have been a little drunk
扁舟何处问渔樵。
With a narrow boat, where could I find the hermits?
行人莫讶归来晚，
Friends, don't be surprised that I come back so late,
万里春风到海潮。
I have seen the great tide and the grand spring breeze.

(b) Style 4: “hermit, rural scenes”

浊酒一杯聊酩酊，
After a cup of unstrained wine,
I have been a little drunk
浮云何处觅仙踪。
I wonder on which cloud I can see the presence of the gods.
迢迢十二峰头月，
The moon above the mount seems farther and farther.
漠漠千山暮霭浓。
The mist among the hill becomes thicker and thicker.

(c) Style 8: “the portrait of hazy sceneries”

Figure 4: Examples generated by style 1,4 and 8 given the same first sentence. The keywords of the three styles are listed for convenience.

Acknowledgments

We would like to thank Chongxuan Li, Huimin Chen, Jiannan Liang, Zhipeng Guo and anonymous reviewers for their insightful comments. This research is funded by the National 973 project (No. 2014CB340501). It is also partially supported by the NExT++ project, the National Research Foundation, Prime Minister's Office, Singapore under its IRC@Singapore Funding Initiative.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

David Barber and Felix Agakov. 2003. The im algorithm: a variational approach to information maximization. In *Proceedings of the 16th International Conference on Neural Information Processing Systems*, pages 201–208. MIT Press.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2016. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Neural Information Processing Systems*.

Pablo Gervás. 2001. An expert system for the composition of formal spanish poetry. *Knowledge-Based Systems*, 14(3-4):181–188.

Jing He, Ming Zhou, and Long Jiang. 2012. Generating chinese classical poems with statistical machine translation models. In *AAAI*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Controllable text generation. *arXiv preprint arXiv:1703.00955*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Tomáš Kočiský, Gábor Melis, Edward Grefenstette, Chris Dyer, Ling Wang, Phil Blunsom, and Karl Moritz Hermann. 2016. Semantic parsing with semi-supervised sequential autoencoders. In *Conference on Empirical Methods in Natural Language Processing*, pages 1078–1087.

Robert P Levy. 2001. A computational model of poetic creativity with neural network as measure of adaptive fitness. In *Proceedings of the ICCBR-01 Workshop on Creative Systems*.

Hisar Manurung. 2004. An evolutionary algorithm approach to poetry generation.

Yael Netzer, David Gabay, Yoav Goldberg, and Michael Elhadad. 2009. Gaiku: Generating haiku with word associations norms. In *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity*, pages 32–39. Association for Computational Linguistics.

Hugo Gonçalo Oliveira. 2012. Poetryme: a versatile platform for poetry generation. *Computational Creativity, Concept Invention, and General Intelligence*, 1:21.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Zhe Wang, Wei He, Hua Wu, Haiyang Wu, Wei Li, Haifeng Wang, and Enhong Chen. 2016. Chinese poetry generation with planning based neural network. *arXiv preprint arXiv:1610.09889*.

Xiaofeng Wu, Naoko Tosa, and Ryohei Nakatsu. 2009. New hitch haiku: An interactive renku poem composition supporting tool applied for sightseeing navigation system. In *International Conference on Entertainment Computing*, pages 191–196. Springer.

Rui Yan. 2016. i, poet: Automatic poetry composition through recurrent neural networks with iterative polishing schema. In *IJCAI*, pages 2238–2244.

Rui Yan, Han Jiang, Mirella Lapata, Shou-De Lin, Xueqiang Lv, and Xiaoming Li. 2013. i, poet: Automatic chinese poetry composition through a generative summarization framework under constrained optimization. In *IJCAI*, pages 2197–2203.

Xiaopeng Yang, Xiaowen Lin, Shunda Suo, and Ming Li. 2017. Generating thematic chinese poetry with conditional variational autoencoder. *arXiv preprint arXiv:1711.07632*.

Xiaoyuan Yi, Ruoyu Li, and Maosong Sun. 2017. Generating chinese classical poems with rnn encoder-decoder. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*, pages 211–223. Springer.

Jiyuan Zhang, Yang Feng, Dong Wang, Yang Wang, Andrew Abel, Shiyue Zhang, and Andi Zhang. 2017. Flexible and creative chinese poetry generation using neural memory. *arXiv preprint arXiv:1705.03773*.

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)*, pages 670–680.