Generating Chinese Ci with Designated Metrical Structure

Richong Zhang[1,2], Xinyu Liu[3], Xinwei Chen[3], Zhiyuan Hu[4], Zhaoqing Xu[2], Yongyi Mao[3] *
[1] SKLSDE, School of Computer Science and Engineering, Beihang University, Beijing, China
[2] Beijing Advanced Institution on Big Data and Brain Computing, Beihang University, Beijing, China
[3] School of Electrical Engineering & Computer Science, University of Ottawa, Ottawa, Ontario
[4] Beijing University of Chemical Technology, Beijing, China

Abstract

Ci is a lyric poetry form that follows highly restrictive metrical structures. This makes it challenging for a computer to compose Ci subject to a specified metrical requirement. In this work, we adapt the CVAE framework to automated Ci generation under metrical constraints. Specifically, we present the first neural model that explicitly encodes the designated metrical structure for Ci generation. The proposed model is shown experimentally to generate Ci with nearly perfect metrical structures.

Introduction

The revival of neural networks, under the new paradigm of deep learning, is impacting many areas of natural language processing. In particular, the generation of natural languages, such as dialogue generation(Moore and Moore 1995), document summarization(Fattah and Ren 2008), machine translation(Bhattacharyya 2015), and poetry generation, has benefited greatly from the methodology of deep learning. Common to these tasks, the deep model must learn a distributed representation of semantics, and be able to map between the semantic representation and a sequence of discrete tokens, i.e., words or characters.

Among these language generation tasks, poetry generation has unique characteristics. That is, not only having implied semantics, poetry is often associated with metrical structures, which specify certain "sounding" rules that a poem must follow. Thus the problem of poetry generation is not only challenged by the need of learning a semantic representation, it is also complicated by the metrical rules to which it must conform. It is worth noting that rule-based learning classically belongs to the symbolist regime of AI, in which rules are usually expressed by certain logical formulas or via some formal language. Neural networks and distributed representations however belong to the connectionist regime of AI, which has been largely disconnected from the symbolist methodology. Thus generating poetry subject to metrical regulation is a perfect example for studying how the two worlds may be combined in a common framework. This has been the motivation behind this work.

*First three authors have equal contribution. Corresponding author: `yymao@eecs.uottawa.ca`

In this work, we consider the problem of generating *Ci*, an ancient Chinese poetry form, subject to their metrical requirements. Comparing with many other genres of poetry, Ci has highly restrictive metrical structures. Specifically, the metrical structure of a Ci is specified by *Cipai*. There are more than 800 Cipai's in the Ci poetry, each specifying at least three kinds of rules: a *rhythmic rule*, a *tonal rule* and a *rhyming rule*. The rhythmic rule dictates the number of lines in the Ci and the varying number of characters in each line, the tonal rule prescribes the tones with which characters at given locations must be pronounced, and the rhyming rule regulates the rhyming pattern. Every Ci must be written for one of these Cipai's and is then subject to the restriction imposed by these metrical rules.

There has been a rich body of literature on automated poetry generation, for example, (Yi, Li, and Sun 2017; Nong 2017; Zhang et al. 2017; Wang et al. 2016b; 2016a; Feigenbaum 2003; Yan 2016; Zhang and Lapata 2014; Yang et al. 2017; Ghazvininejad et al. 2016; Oliveira et al. 2014; Rashel and Manurung 2014). These existing methods can be divided into "template-based" approaches and "learning-based" approaches. The template-based approaches, such as (Oliveira 2012; Oliveira et al. 2014; Rashel and Manurung 2014), usually involve designing a set of templates and certain scoring functions, in a process often requiring a significant amount of intervention from human experts. The templates are then used to generate a set of candidate poems, which are further evaluated by the scoring function. The candidates poem with top scores are then returned as the output. The learning-based approaches, such as (Yi, Li, and Sun 2017; Zhang et al. 2017; Wang et al. 2016b; 2016a; Yan 2016; Zhang and Lapata 2014; Yang et al. 2017), on the other hand, rely on designing a probabilistic or neural network model and training the model on a large poetry corpus. The trained models are then used to generate poems.

In general, the template-based approaches are capable of explicitly importing the metrical rules in the design of templates, but the quality of the generated poems are highly dependent of the poetry expertise of the designer. The learning-based approaches on the other hand aim at "learning" the poetry expertise from the corpus automatically and rely less on human experts. However, they largely ignore the *known* metrical structure of the poetry and hope that the model can be sufficiently expressive and learnable to extract the metrical rules. To what extent this is possible is questionable, since it not only depends on the capacity of the model but also requires a sufficient amount of training data. This issue is arguably further magnified if the poetry to be generated has highly restrictive metrical structures, such as Ci.

With the learning-based perspectives, the philosophy of this work is to directly encode the known metrical structures of Ci into neural networks. In this paper, we show that the Conditional Variational Auto-Encoder (CVAE) framework (Sohn, Lee, and Yan 2015) can be naturally adapted to generating Ci with a designated Cipai. Specifically, we present the Metrically Restricted Ci Generation (MRCG) model using this framework. In this model, the metrical structure dictated by a Cipai is explicitly encoded and supplied to the CVAE model as the "condition". Through experiments, we demonstrate that the MRCG model is capable of generating excellent metrical structure required by the Cipai. Our results also suggest that rules and prior knowledge can be integrated into a neural network model, and that a unification of the connectionist and symbolist regimes in the construction of learning systems can be made possible.

Ci and Their Metrical Structures

Ci, also known as *Shiyu* or *Changduanju*, is a lyric poetry form in ancient Chinese poetry. Although having its root in much earlier ancient Chinese poetry, such as Shijing (about 500 BC), Ci was known to flourish in the Tang Dynasty (618-907 AD). It was further developed and reached its pinnacle in the Song Dynasty (960-1279 AD). Ci of the Song Dynasty is widely recognized as one of the highest achievements in the classical Chinese literature.

Historically, Ci were written as song lyrics for a set of popular tunes. For this reason, each such tune, called a *Cipai*, regulates strictly a unique metrical pattern which the Ci must follow. To date, over 800 Cipai's have been used to compose Ci, where every Cipai prescribes a metrical pattern by simultaneously insisting at least the three kinds of rules below.

Rhythmic Rule. In general, the number of characters in each line (i.e. sentence or clause) and the number of lines may vary. The *rhythmic rule* of a Cipai specifies strictly the length of each line in the Ci and the total length of the Ci. Depending on the Cipai, the number of characters in a Ci can range from tens to about a hundred, and the number of lines in a Ci can range from a few to tens.

Tonal Rule. The ancient Chinese language, particularly during the Tang and Song Dynasties, uses five different tones. Every character was pronounced in one the five tones. The five tones are further classified into two categories, known as *Ping* and *Ze*[1]. The *tonal rule* of a Cipai dictates the Ping/Ze pattern of the Ci. More specifically, for most character locations in the Ci, the tonal rule specifies whether the location must be filled with a character having the Ping tone or filled with a character having the Ze tone. In other character locations, the tonal rule insists no tonal constraint, namely, the location can be filled with a character having any tone.

Rhyming Rule. Unlike the European languages, in which the fundamental pronunciation units are consonants and vowels, the fundamental units in Chinese pronunciation are *initials*, and *finals*. For the purpose of this paper, there is no harm to interpret initials simply as the Chinese-equivalent of consonants and finals as the Chinese-equivalent of vowels. A *syllable* in Chinese (particularly Mandarin) can be most naturally pronounced by pronouncing an initial immediately followed by a final. Almost all Chinese characters are pronounced in a single syllable. In Ci, lines that have their ending characters pronounced with the same final are said to *rhyme* with each other, and each of the ending characters is called a *rhyming foot*. In the lines that rhyme, the common final pronounced in the rhyming feet is referred to as the *Yunshe*. Not only between consecutive sentences, in Ci, rhyming is also allowed between lines that are one or a few lines apart. A group of consecutive lines that rhyme on the same Yunshe (allowing non-rhyming lines within) may be referred to as a *rhyming group*. In some case, all lines in a Ci form a single rhyming group; in other cases, a Ci may be partitioned into a few rhyming groups, each rhyming on a different Yunshe. The *rhyming rule* of a Cipai may also specify the partition of the lines in the Ci into rhyming groups, and in each group, which lines must rhyme. What Yunshe to use in each rhyming group is not enforced by the rhyming rule. According to the ancient Chinese pronunciation, there are in total 16 finals, and hence 16 Yunshe's.

We note that beyond these three kinds of rules, there are additional miscellaneous and often subtle metrical requirements that a Ci must satisfy. Vary with Cipai, those finer metric prescriptions are ignored in this paper.

Comparing with other poetry forms, such as the Qiyan and Wuyan in Chinese poetry, Haiku in Japanese poetry, or Couplets and Fourteeners in English poetry, the highly restrictive metrical structure dictated by a Cipai arguably imposes greater difficulties in the composition of Ci.

Applying the CVAE Framework

Let $c = (s_1, s_2, \ldots, s_L)$ be a Ci with L lines, where line l is denoted by s_l. In particular, each s_l is a sequence of $N(l)$ words $(w_{l,1}, w_{l,2}, \ldots, w_{l,N(l)})$. Let r denote the Cipai regulating the metrical structure of c. We will postpone to a later section the precise form of r. Throughout the paper, we use a lower-cased letter, e.g. c, to express the *realization* of a random variable, and the random variable itself is denoted by its corresponding upper-cased letter, e.g., C.

A natural probability model for c with a designated Cipai r is given below.

$$p_{C|R}(c|r) := \int p_{C|RZ}(c|r,z)p_Z(z)dz, \qquad (1)$$

where Z is a *latent* semantic representation of Ci C. Note that in (1), we assume that the Ci C depends on its latent semantics Z and its metrical structure R jointly. Additionally, it is assumed that Z does not depend on the Cipai R. This latter assumption is justified since for each Cipai, Ci having arbitrary semantics can be written, and the metrical structure insisted by a Cipai imposes no semantic restrictions.

[1]Modern Chinese only uses four tones with one of the ancient tone lost. But the classification of modern Chinese characters into the *Ping* and *Ze* classes largely remains the same as that in ancient Chinese.

Cipai：忆江南（Yi Jiang Nan）

Rhythmic Rule:	3 characters，5 characters。 7 characters ， 7 characters 。 5 characters?
Ci:	江南好，风景旧曾谙。日出江花红胜火，春来江水绿如蓝。能不忆江南?
Tonal Rule:	0 + - ， 0 - - + + 。 0 - 0 + + - - ， 0 + + - - + + 。 0 - - + + ?
Rhyming Rule:	- - - ， - - - - x 。 - - - - - - - ， - - - - - - x 。 - - - - x ?
Translation:	Fair Southern shore, with scenes I adore. At sunrise riverside flowers redder than fire. In spring green waves grow as blue as sapphire. Which I cannot but admire.

Figure 1: An example of Ci "Yi Jiang Nan", with translation taken from (Yuanchong 2005)

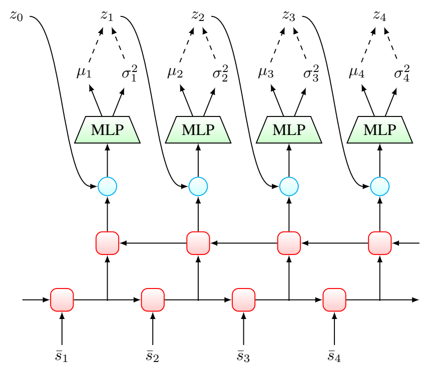

Figure 2: Context embedding and generation of latent semantic representation in the encoder. Round-corner box: the function $\mathbf{GRU}(\cdot)$. Circle: the function $\mathbf{concat}(\cdot)$.

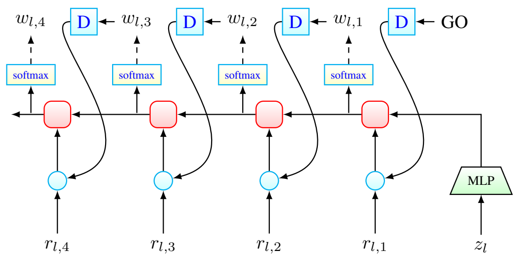

Figure 3: Generating line s_l in the decoder. Round-corner box: the function $\mathbf{GRU}(\cdot)$. Circle: the function $\mathbf{concat}(\cdot)$.

We note that the term *semantics* in this paper refers to the *meaning* of a Ci in a broad sense. It may include the literal meaning, connotation, mood, sentiment, emotion, impression, imagery, conception or any other aspects of semantics.

The integration in (1) makes it in general intractable to learn such a model through log-likelihood. In the CVAE framework(Sohn, Lee, and Yan 2015), the log-likelihood in (1) is bounded by the following *variational lower bound*, for any arbitrary conditional distribution $q_{Z|CR}$:

$$\log p_{C|R}(c|r) \geq -\,\mathrm{KL}(q_{Z|CR}(\cdot|c,r)||p_Z)+ \\ \mathbb{E}_{z\sim q_{Z|CR}}\left\{\log p_{C|RZ}(c|r,z)\right\}$$

Similar to an earlier argument, we assume that the semantic representation Z is independent of the metrical structure R given Ci C. Thus the $q_{Z|CR}$ reduces to $q_{Z|C}$. Usually $p_{C|RZ}$ is called the *decoder*, and we will use Θ to denote its parameters; $q_{Z|C}$ is called the *encoder*, and we will use Φ to denote its parameters. The prior distribution p_Z in this paper will take a fixed distribution.

Define the "KL loss" and the "reconstruction loss" by

$$\begin{aligned} \ell_{\mathrm{KL}}(c,r;\Phi) &:= \mathrm{KL}(q_{Z|CR}(\cdot|c,r)||p_Z) \\ \ell_{\mathrm{rec}}(c,r;\Theta,\Phi) &:= -\mathbb{E}_{z\sim q_{Z|CR}}\left\{\log p_{C|RZ}(c|r,z)\right\}. \end{aligned}$$

Let $\mathcal{D}$ denote a set of training examples, each being a pair (c,r). Then the learning of model (1) can be solved by maximizing the variational lower bound or minimizing the loss function

$$\mathcal{L}(\Theta,\Phi) := \sum_{(c,r)\in\mathcal{D}} \left\{\ell_{\mathrm{KL}}(c,r;\Phi)+\ell_{\mathrm{rec}}(c,r;\Theta,\Phi)\right\} \qquad (2)$$

We note that this framework is in general applicable to generating any form of poetry beyond Ci. We now present a model specific for Ci generation.

Proposed MRCG Model

We will use $\mathcal{N}(x;\mu,\kappa)$ to denote a multi-variate Gaussian density function (in variable x) with mean vector μ and covariance matrix κ. The processing in a GRU network(Chung et al. 2014) is denoted by function $\mathbf{GRU}(\cdot)$, namely, the state h_t in the GRU is updated by

$$h_t = \mathbf{GRU}(h_{t-1}, x_t)$$

where x_t denotes the input of the GRU at time t. We also call the dimension of h_t the dimension of the GRU. For the exact composition of the function $\mathbf{GRU}(\cdot)$, the reader is referred to (Chung et al. 2014). Unless otherwise stated, the initial state of every GRU used in this paper is set to be the all-zero vector by default.

We use $\mathbf{MLP}(\cdot)$ to denote a function that maps an input vector to an output vector via a *multi-layer perceptron* network. The notation $\mathbf{concat}(\cdot)$ refers to the operation that concatenates two vectors into a single vector. Notation $\mathbf{softmax}(\cdot)$ refers to the standard soft-max transform

(Nasrabadi 2007) converting a score vector into a probability vector having the same length.

For a Ci $c = (s_1, s_2 \dots, s_L)$ containing L lines, we let its latent semantic representation z take the form $z := (z_1, z_2, \dots, z_L)$, where each z_l is a vector in $\mathbb{R}^K$, corresponding to the line s_l. The prior distribution p_Z is chosen such that each Z_l is independently distributed according to $\mathcal{N}(z_l; 0, \mathbf{I})$ on $\mathbb{R}^K$. Here $\mathbf{I}$ denotes the identity matrix.

Encoder Model Following a similar architecture as in (Serban et al. 2016), the proposed encoder successively encodes the Ci c into three levels of representations, i.e. character embeddings, line embeddings, and context embeddings (Figure. 2). The latent semantic representation is then generated. We now give a precise description of the encoder.

Character Embedding Let $\mathcal{V}$ denote the vocabulary of all characters. Each character in $\mathcal{V}$ can then be identified with a *one-hot* vector in $\mathbb{R}^{|\mathcal{V}|}$. Let $\mathbf{D}$ be a $K^{\mathrm{w}} \times |\mathcal{V}|$ matrix. The character embedding $\overline{w}_{l,j}$ of the j^{th} character $w_{l,j}$ in the l^{th} line is then a K^{w}-dimensional vector constructed by

$$\overline{w}_{l,j} := \mathbf{D} w_{l,j} \tag{3}$$

Line Embedding The embedding $\overline{s}_l$ of line s_l is a vector constructed by passing the embeddings of the characters in s_l as the input sequence to a GRU network. More specifically, let $h^{\mathrm{s}}_{l,t}$ be the state variable of the GRU network that encodes line l. The computation of the state $h^{\mathrm{s}}_{l,t}$ is via

$$h^{\mathrm{s}}_{l,j} := \mathbf{GRU}(h^{\mathrm{s}}_{l,j-1}, \overline{w}_{l,j}). \tag{4}$$

We then take the final state of the GRU network as the embedding $\overline{s}_l$ for line s_l, namely,

$$\overline{s}_l := h^{\mathrm{s}}_{l,N(l)} \tag{5}$$

The dimension of the GRU is denoted by K^{s} and it parameters are denoted by Φ^{s}.

Context Embedding For each s_l, a vector u_l with dimension K^{u}, referred to as the *context embedding* of s_l, is constructed. These embeddings are constructed using a stacked two-layer GRU network. Both layers of the network have dimension K^{u}. The states $h^{\mathrm{u},1}_l$ of the first layer is updated by

$$h^{\mathrm{u},1}_l := \mathbf{GRU}(h^{\mathrm{u},1}_{l-1}, \overline{s}_l) \tag{6}$$

These states are then fed into the second layer as input, but in reverse order. The state $h^{\mathrm{u},2}_l$ of the second layer is computed recursively by

$$h^{\mathrm{u},2}_l := \mathbf{GRU}(h^{\mathrm{u},2}_{l-1}, h^{\mathrm{u},1}_{L+1-l}) \tag{7}$$

The states of the second layer are taken as context embeddings, i.e., the context embedding of s_l is

$$u_l := h^{\mathrm{u},2}_{L-l+1} \tag{8}$$

Latent Semantic Representation Overall the encoder model $q_{Z|C}$ takes the following form.

$$q_{Z|C}(z_1, z_2, \dots, z_L|c) := \prod_{l=1}^{L} q_{Z_l|CZ_{l-1}}(z_l|c, z_{l-1}), \tag{9}$$

where z_0 is set to the all-zero vector in $\mathbb{R}^K$. Each $q_{Z_l|CZ_{l-1}}$ in (9) is defined as the

$$q_{Z_l|CZ_{l-1}}(z_l|c, z_{l-1}) := \mathcal{N}(z_l; \mu_l, \mathrm{diag}(\sigma_l^2)) \tag{10}$$

where both μ_l and σ_l^2 are vectors in $\mathbb{R}^K$ and are computed from u_l and z_{l-1} using an MLP:

$$\mathbf{concat}(\mu_l, \log \sigma_l^2) := \mathbf{MLP}(\mathbf{concat}(u_l, z_{l-1})). \tag{11}$$

The MLP is structured to contain one hidden layer with PReLu activation(He et al. 2015) and one linearly-activated output layer.

Decoder Model We will use $\mathcal{T} := \{+, -, *\}$ to denote the set of possible tones. Here $+$ and $-$ denote Ping and Ze respectively and $*$ denotes "NO SOUND", namely, the tone for the punctuation marks, the unknown or rare ancient characters whose tone can not be determined. Thus the one-hot vectors in $\mathbb{R}^3$, referred to as the *tone vectors*, can be used to represent all three considered tones. For any character w, we will use $a(w)$ to denote its tone vector.

Similarly, recalling that there are 16 Yunshe's, we will use $\mathcal{Y} := \{1, 2, \dots, 16, *\}$ to denote the set of all possible Yunshe's, where the symbol $*$ also refers to "NO SOUND". Thus, the one-hot vectors in $\mathbb{R}^{17}$, called the *Yunshe vectors*, can be used to represent all 17 considered Yunshe's. We use $b(w)$ to denote Yunshe vector of character w.

We note that $a(w)$ (resp. $b(w)$) can be regarded as a probability vector which puts probability one on one of the tones (resp. on one of the Yunshe's).

Encoding Metrical Structure For a given Cipai r which prescribes a Ci to have L lines, we represent it as a sequence $(r_1, r_2, \dots, r_L)$. Here r_l specifies the metrical structure for line s_l. In addition, r_l is as a sequence $(r_{l,1}, r_{l,2}, \dots, r_{l,N(l)})$, where $r_{l,j}$ specifies the rule that character $w_{l,j}$ must follow. We now explain $r_{l,j}$ and its encoding.

First we construct a *tone-rule vector* $\widetilde{a}_{l,j}$, which is vector in $\mathbb{R}^3$ representing a distribution over the set $\mathcal{T}$ of all possible tones. Specifically, if $r_{l,j}$ specifies that $w_{l,j}$ must take tone "+" (resp. "$-$"), then $\widetilde{a}_{l,j}$ is the one-hot vector for "+" (resp. for "$-$"). If $r_{l,j}$ specifies that $w_{l,j}$ must be a punctuation mark, then $\widetilde{a}_{l,j}$ is the one-hot vector for "$*$". If $r_{l,j}$ does not specify the tone of $w_{l.j}$, then $\widetilde{a}_{l,j}$ is the vector $[.5, .5, 0]^T$, which puts probability 0.5 on "+" and "$-$", and puts probability 0 on "$*$".

We will also need to construct a *rhyme-rule vector* $\widetilde{b}_{l,j}$, which is a vector in $\mathbb{R}^{17}$ representing a distribution over the set $\mathcal{Y}$ of all possible Yunshe's.

If $r_{l,j}$ suggests that $w_{l,j}$ is a character (i.e., not a punctuation mark) and not a rhyming foot, $\widetilde{b}_{l,j}$ is taken as the empirical distribution, say, $\widetilde{p}$, over the 16 Yunshe's in the dataset. [2]

If $r_{l,j}$ suggests that $w_{l,j}$ is a punctuation mark, then $\widetilde{b}_{l,j}$ is taken as the one-hot vector for "$*$".

If $r_{l,j}$ dictates that $w_{l,j}$ is a rhyming foot in a rhyming group, then $\widetilde{b}_{l,j}$ need to be specified with the rhyme-rule vectors $\widetilde{b}_{l',j'}$'s for all other rhyming feet $w_{l',j'}$'s in the same rhyming group. In this case, for this rhyming group, a random Yunshe Y is drawn from $\widetilde{p}$, and the rhyme-rule vector is set to the one-hot vector for Y.

With $\widetilde{a}_{l,j}$ and $\widetilde{b}_{l,j}$ constructed, we can then construct $r_{l,j}$ as a vector in $\mathbb{R}^{20}$, defined as

$$r_{l,j} := \mathbf{concat}(\widetilde{a}_{l,j}, \widetilde{b}_{l,j}). \tag{12}$$

We note that although this scheme of constructing $r_{l,j}$ entails no difficulty in testing (i.e., generating Ci), it does result in high complexity (due to sampling in the construction of $\widetilde{b}_{l,j}$). To resolve this issue, we actually take advantage of the fact the Ci is available at the training time and take $\widetilde{a}_{l,j} = a(w_{l,j})$ and $\widetilde{b}_{l,j} = b(w_{l,j})$ during training.

We note that although $\widetilde{a}_{l,j}$ and $\widetilde{b}_{l,j}$ are designed for encoding the tonal and rhyming rules, they also encode the rhythmic rule. This is because $\widetilde{a}_{l,j}$ and $\widetilde{b}_{l,j}$ also contains indicators for "NO SOUND". With the model properly learned, these indicators force the punctuation marks to arise at the correct locations and impose the correct rhythmic parsing.

Generating Ci Overall the decoder model $p_{C|RZ}$ (Figure. 3) is assumed to take the form

$$p_{C|RZ}(c|r,z) := \prod_{l=1}^{L} p_{S_l|R_lZ_l}(s_l|r_l,z_l). \tag{13}$$

Further each $p_{S_l|R_lZ_l}(s_l|r_l,z_l)$ factors as

$$p_{S_l|R_lZ_l}(s_l|r_l,z_l) = \prod_{j=1}^{N(l)} p_{W_{l,j}|W_{l,j-1}R_lZ_l}(w_{l,j}|w_{l,j-1},r_l,z_l) \tag{14}$$

The model for $p_{W_{l,j}|W_{l,j-1}R_lZ_l}$ is constructed via another GRU network. Specifically, we let the initial state $h^{\mathrm{w}}_{l,0}$ of the network be obtained by

$$h^{\mathrm{w}}_{l,0} := \mathbf{MLP}(z_l) \tag{15}$$

Let the state $h^{\mathrm{w}}_{l,j}$ of the GRU be computed by

$$h^{\mathrm{w}}_{l,j} := \mathbf{GRU}(h^{\mathrm{w}}_{l,j-1}, \mathbf{concat}(\mathbf{D}w_{l,j-1}, r_{l,j})) \tag{16}$$

[2]When using the trained MRCG to generate a Ci, we find it is beneficial to make the $\widetilde{b}_{l,j}$ "peakier". Then we actually draw M samples from $\widetilde{p}$ and use the empirical distribution of the M samples as $\widetilde{b}_{l,j}$.

where $w_{l,0}$ is the special token character "GO" in the character vocabulary $\mathcal{V}$. We will denote the dimension of this GRU by $K^{\mathrm{w}'}$.

Let V be a $|\mathcal{V}| \times K^{\mathrm{w}'}$ matrix. The distribution $p_{W_{l,j}|W_{l,j-1}R_lZ_l}$ is then computed, as a vector, by

$$p_{W_{l,j}|W_{l,j-1}R_lZ_l} := \mathbf{softmax}(Vh^{\mathrm{w}}_{l,j}). \tag{17}$$

At this end, our MRCG model is completely specified via Equations (2) to (17).

Training

Mini-batched Stochastic Gradient Descent (SGD) can be used to minimize the loss function in (2). Specifically the "reparametrization trick" (Kingma and Welling 2013) is used to generate the latent semantic representations, as is standard.

It is observed in the literature that training of CVAE or VAE model may suffer from the issue of "KL vanishing" (namely, that the KL loss tends to zero) when the decoder has high capacity(Bowman et al. 2015). We adopt the KL clamping approach presented in (Kingma et al. 2016) to resolve this issue, where when the KL loss is below a prescribed threshold, it is clamped to the threshold and no longer decreased with SGD.

Experimental Study

We now study the performance of the proposed MRCG model. Ideally we would like to compare MRCG with some recent art in Ci generation. It is unfortunate however that the papers presenting those models do not include sufficient implementation details, nor did the authors make available their source code. As such, we instead implement two baseline models, one being a Seq2Seq model (Sutskever, Vinyals, and Le 2014) and the other being an attention model(Bahdanau, Cho, and Bengio 2014) for comparison. We note that these baselines underlie most of the recent neural models for poetry generation and should be reasonably representative.

Datasets and experimental setup

There is no standard Ci corpus to date. We extract a dataset from a Chinese poetry website(Yun 2017). The dataset contains 82,724 Ci's, written for 818 Cipai's. The distribution of the number Ci's per Cipai roughly follows an exponential distribution. Specifically, 95% of the corpus are contributed by the most popular 314 Cipai's, and the most popular 20 Cipai's contribute to about 45% of the corpus. The average number of lines per Ci is about16. The number of characters per line is about 7 on average. The dataset is split into the training set and the testing set as follows. For each Cipai, we randomly select 5% of its Ci's to assemble the testing set. For the Cipai's having less than 20 Ci's, they are not included in the testing set. In total, we have 3,797 Ci's in the testing set and 78,927 in the training set.

In our implementation of MRCG, we choose the latent semantic dimension $K = 64$. Other dimensions are chosen as $K^{\mathrm{w}'} = K^{\mathrm{w}} = K^{\mathrm{s}} = 128$, and $K^{\mathrm{u}} = 256$.

We also implemented a reduced version of MRCG, which we refer to as MRCG^-, with the same hyper-parameter settings. The difference between MRCG^- and MRCG is that in MRCG^- the metrical information is completely removed.

Both the Seq2Seq model and the attention model are built using the tensorflow NMT tool (Luong, Brevdo, and Zhao 2017). In both baselines, bi-directional GRU layers are used in the encoder, and single-directional GRU layers is used as the decoder. In order to compensate for the structural disadvantages of these baselines relative to MRCG, we used multiple such layers in the encoder and decoder so that the resulting model has a similar number of parameters to that of MRCG.

We remark that the MRCG model and the two baselines differ in a fundamental way. That is, MRCG is a model conditioned on a designated Cipai, whereas the two baselines are models conditioned on a Ci. The objective of MRCG is to *generate* a Ci complying with the Cipai's requirement, whereas the objective of the baselines is to *reproduce* the Ci that is input to the model. For this reason, we design two test protocols.

Blind Generating. This test is only for MRCG and MRCG^-. In this test, a random Cipai is passed to the model as input, and a latent semantic representation is drawn from the *prior* p_Z to generate an output Ci.

Guided Generating. This test applies to all compared models. For the baselines, they take a random input Ci from the testing set as input and attempt to reproduce it at the output. For MRCG and MRCG^-, they take as input the same random Ci and its Cipai. The latent semantic representation is drawn from the approximating posterior $q_{Z|CR}$, which depends on both inputs.

For both tests, beam search is used to generate Ci from the model's predictive distribution.

Metrical Performance

We use three metrics to quantify how well a generated Ci meets the Cipai requirement.

Length Accuracy. Given a Ci to be evaluated, we compare the length of each line with that specified by the rhythmic rule of the Cipai. The *length correctness* of the Ci is then defined as the percentage of lines in the Ci that have the correct length. The *length accuracy* is the average of length correctness over all generated Ci's.

Tone Accuracy. Given a generated Ci and for each character within, we check if the tone of the generated character satisfies what is dictated by the tonal rule of the Cipai. The *tone correctness* of the Ci is then defined as the percentage of the characters in the Ci that have the correct tone. The *tone accuracy* is the average of tone correctness over all generated Ci's.

Rhyme Accuracy. Given a generated Ci and for each rhyming group within, the *rhyme correctness* of the rhyming group is defined as the fraction of rhyming foot locations that have the correct Yunshe. Specifically, the correct Yunshe is taken as the majority Yunshe in the rhyming group. The *rhyme accuracy* is the average of rhyme correctness over all rhyming groups in a Ci and then further averaged over all generated Ci's.

Table 1: Metrical Performances

Model	Blind Generation			Guided Generation		
	Length	Tone	Rhyme	Length	Tone	Rhyme
Seq2seq	-	-	-	27.1	55.0	29.8
Attn	-	-	-	28.2	56.3	26.7
MRCG^-	35.49	59.43	33.34	41.96	66.21	39.77
MRCG	**99.21**	**92.03**	**96.87**	**99.37**	**93.71**	**98.28**

Table 1 shows the metrical performances of the compared models under the three metrics. Note in the table, MRCG, under either guided or blind generation, demonstrates excellent accuracy scores, outperforming the baselines by a huge margin. This validates the effectiveness of MRCG in following the designated metrical structure. It also suggests that without explicitly designing the model to handle Cipai requirements as in the baselines, it is much more difficult for the model to learn the metrical rules. This is at least the case when the training data isn't abundant.

MRCG shows a tonal accuracy slightly inferior to its other two metrical performances. We believe that this is due to polyphones in Chinese. In Chinese some characters have multiple pronunciations, and hence multiple tones. In our dataset, we set the tone of each character to its most popular one. This introduces training noise, thereby reducing the tone accuracy. It is worth noting however that in Chinese, a character having multiple pronunciations usually has the same Yunshe across its different pronunciations. This is the reason that the rhyme accuracy is not affected as much. We expect that carefully looking after polyphones in the training data will improve the tonal accuracy.

Semantic Performance

Although the main focus of this work is to integrate known metrical structure in the generation of Ci, it is important that such an integration does not result in deteriorated semantics. To verify this, we carry out several semantic tests.

Absolute Semantics Test (AST). Following the approach of (He, Zhou, and Jiang 2012; Jiang and Zhou 2008), this test is conducted as follows. A random Ci is generated from a model. For every two consecutive lines therein, say $(\widehat{s}_1, \widehat{s}_2)$, we find from a Ci database Shixuehanying (Liu 1735) 10 lines that are the most similar to $\widehat{s}_1$. For each of these 10 lines, we collect its next line in the database into a set S_2, then the BLEU score of $\widehat{s}_2$ against S_2 is computed. After repeating this experiments, the average BLEU score is used to evaluate the model.

Relative Semantics Test (RST). We devise this test to evaluate, under guided generation, whether the outputs of a model preserve the similarities and differences between its inputs. Specifically, a pair of Ci's (c_1, c_2) are drawn at random from the testing set, both of which are used as an input to the model. Let $(\widehat{c}_1, \widehat{c}_2)$ be their respective outputs from the model. We then compute the BLEU score S of c_1 against c_2 and the BLEU score $\widehat{S}$ of $\widehat{c}_1$ against $\widehat{c}_2$. After this experiment is repeated many times, the Pearson Correlation Coefficient between S and $\widehat{S}$ is used to evaluate the model.

Cipai：长相思（Chang Xiang Si）	
钿钗头	The pretty hairpin I wear
等闲愁	With loneliness and boredom
泪滴红蕤衬玉钩	Tears drop on the moon-lit pillow
夜来微湿秋	Lightly drenching this night of autumn
淡墨淋	Let my pen and ink
东晋游	Run through the Dynasty of Jin
百啭莺吭春梦稠	In the dream of romance, let thousands of doves chirp
秋千闲倚楼	Where on a leisureful balcony, I play on a swing

Cipai：清平乐（Qing Ping Yue）	
消魂销尽	Have you lost your mind
昨梦沈郎不	Dreaming of Mr. Shen last night?
莫惜飘茵随逝日	Don’t pity the green grass withering in the wind of time
泪雨潇湘月明	Tears like rain, you linger in the moonlight
只愁听玉钩花	Sadness when you listen to an old song
白苹桥外啼鸦	Resonates with the raven, crying on the bridge
别离恨愁无迹	Leave behind the pain of parting
吟怀谁把年华	For an old memory, who would sacrifice youth, beautiful and shining?

Figure 4: Two example Ci’s generated by MRCG

Table 2: Semantics Test Results

Model	Objective Tests		Human Evaluation			
	AST	RST	Flu	The	Aes	All
Seq2Seq	**0.242**	0.358	**2.65**	**2.8**	2.26	2.33
Attn	0.221	0.323	2.53	2.72	2.25	2.21
MRCG-	0.235	**0.541**	2.38	2.41	2.34	2.26
MRCG	0.229	0.529	2.59	2.68	**2.54**	**2.45**

Human Evaluation. A total of 12 Human examiners are invited to evaluate 25 Ci’s randomly generated by each model (without revealing the identity of the model). Specifically, fluency (Flu), theme consistency (Thm), aesthetics (Aes) and overall (All) performance are evaluated using scores $\{1, 2, 3, 4\}$, where the examiners are told to calibrate the score of 1 to “poor” and the score of 4 to “expert level”.

The results of these tests are shown in Table 2. From these results, it can be seen that MRCG performs comparably to the compared models. This suggests that the proposed encoding and integration of metrical structures bring no negative impact on semantics.

In fact, by inspecting a large number of Ci generated by MRCG (guided or blind), we observe that most generated of them have coherent sentiment and smooth grammar. Many of them contain some excellent lines whose literacy levels are comparable to those in the classical Ci literature. Figure 4 contains a demonstration of two Ci’s generated blindly by MRCG, together with their English translations.

It must be noted however that overall an educated Ci reader would still be able to distinguish the Ci’s generated by MRCG from those in the classical Ci literature, due to the limitations of MRCG that we will present next. For this reason, we believe that MRCG will not pass the Turing test. This will also be the case for the baselines, since, due to their poor metrical performances, they would fail obviously at the metrical level.

Limitations of MRCG

As discussed above, we believe that MRCG represents the state of the art in the neural models for Ci generation. Nonetheless, our experiments suggest that Ci generated by MRCG still exhibits a visible gap from those written by human writers.

First, in some cases, certain implicit rhythmic rules are not obeyed in the generated Ci. For example, suppose that a particular line in some Cipai is to contain 5 characters in the format of $(AB)(CD)(E)$, where a pair of parentheses groups the characters forming a phrase. But occasionally the line generated by MRCG may have the structure $(A)(BC)(DE)$. Note that such a rule is an implicit convention in Ci composition for that Cipai, and we did not encode the rule in the MRCG model. As such, MRCG model can only learn such a rule from the training examples. Depending on the number of training examples for this rhythmic pattern and that of the confusing examples[3], the learning of such a rhythmic rule may not be very effective. We however expect that such a problem can be resolved to a good extent by fine-tuning the MRCG model so that such rules are explicitly encoded.

Another limitation we observe in MRCG is on the semantics side. Although MRCG is capable of expressing moods, sentiments and emotions very well by creating imageries containing the essential related elements, the model appears to be weak in generating a rich story with logical coherence. Specifically, in some Ci’s generated by MRCG, the transition of ideas appears to be largely governed by a smooth evolution of sentiments, rather than by a smooth development of an idea or story that can be well reasoned logically. Thus these Ci, particularly the longer ones within, may appear full of emotions but lacking ideas or stories.

Concluding Remarks

Chinese Ci is among the poetry forms that have the most restrictive metrical structures. This makes computer composition of Ci a great challenge. In this paper, we show that the CVAE framework can be naturally adapted to the metri-

[3]Depending on the Cipai, a line containing 5 characters may take both the format of $(AB)(CD)(E)$ and the format of $(A)(BC)(DE)$ format. Thus the model must learn to distinguish the two formats.

cally restricted CI generation. Based on this framework, we propose the MRCG model which explicitly encodes a designated metrical structure. We show via experiments that the Ci generated from this model satisfy the desired metrical restriction nearly perfectly and have good semantics.

To the best of our knowledge, the MRCG model is the first model that explicitly encodes the metrical restriction in a neural model. The methodology employed in MRCG may be seen as a great demonstration that the rule-based learning in the symbolist paradigm and the neural network approaches in the connectionist paradigm can be seamlessly integrated into the same distributed representation framework. We believe that such a methodology can be extended to generating poetries with other metrical constraints and, beyond this context, to integrating rules into a neural network based learning model.

By now, we have an excellent solution to generating Ci with a desired metrical structure. The bottleneck in Ci generation, or more generally in poetry generation, then remains on the semantics side. How to generate poetry with "human-level semantics" is still a grand challenge in computational linguistics. Significant advances in this direction, we believe, will rely on the development of new natural language understanding models, particularly those capable of representing *reasoning* at a fundamental level.

Acknowledgments

This work is supported partly by China 973 program (2015CB358700), by the National Natural Science Foundation of China (61772059,61421003), and by the Beijing Advanced Innovation Center for Big Data and Brain Computing and State Key Laboratory of Software Development Environment.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *Computer Science*.

Bhattacharyya, P. 2015. *Machine translation*. CRC Press.

Bowman, S. R.; Vilnis, L.; Vinyals, O.; Dai, A. M.; Jozefowicz, R.; and Bengio, S. 2015. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*.

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Fattah, M. A., and Ren, F. 2008. Automatic text summarization. *World Academy of Science, Engineering and Technology* 37:2008.

Feigenbaum, E. A. 2003. Some challenges and grand challenges for computational intelligence. *Journal of the ACM (JACM)* 50(1):32–40.

Ghazvininejad, M.; Shi, X.; Choi, Y.; and Knight, K. 2016. Generating topical poetry. In *EMNLP*, 1183–1191.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, 1026–1034.

He, J.; Zhou, M.; and Jiang, L. 2012. Generating chinese classical poems with statistical machine translation models. In *AAAI*.

Jiang, L., and Zhou, M. 2008. Generating chinese couplets using a statistical mt approach. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, 377–384. Association for Computational Linguistics.

Kingma, D. P., and Welling, M. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Kingma, D. P.; Salimans, T.; Jozefowicz, R.; Chen, X.; Sutskever, I.; and Welling, M. 2016. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, 4743–4751.

Liu, W. 1735. *ShiXueHanYing*.

Luong, M.; Brevdo, E.; and Zhao, R. 2017. Neural machine translation (seq2seq) tutorial. *https://github.com/tensorflow/nmt*.

Moore, J. D., and Moore, J. D. 1995. *Participating in explanatory dialogues: interpreting and responding to questions in context*. MIT press Cambridge, MA.

Nasrabadi, N. M. 2007. Pattern recognition and machine learning. *Journal of electronic imaging* 16(4):049901.

Nong, D. X. L. 2017. Dao Xiang Poems Generator. `http://www.poeming.com/web/index.html/`. [Online; accessed 20-November-2017].

Oliveira, H. G.; Hervás, R.; Díaz, A.; and Gervás, P. 2014. Adapting a generic platform for poetry generation to produce spanish poems. In *ICCC*, 63–71.

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

Rashel, F., and Manurung, R. 2014. Pemuisi: a constraint satisfaction-based generator of topical indonesian poetry. In *ICCC*, 82–90.

Serban, I. V.; Sordoni, A.; Bengio, Y.; Courville, A. C.; and Pineau, J. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, volume 16, 3776–3784.

Sohn, K.; Lee, H.; and Yan, X. 2015. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, 3483–3491.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *NIPS*, 3104–3112.

Wang, Q.; Luo, T.; Wang, D.; and Xing, C. 2016a. Chinese song iambics generation with neural attention-based model. *arXiv preprint arXiv:1604.06274*.

Wang, Z.; He, W.; Wu, H.; Wu, H.; Li, W.; Wang, H.; and Chen, E. 2016b. Chinese poetry generation with planning based neural network. *arXiv preprint arXiv:1610.09889*.

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

Yang, X.; Lin, X.; Suo, S.; and Li, M. 2017. Generating thematic chinese poetry with conditional variational autoencoder. *arXiv preprint arXiv:1711.07632*.

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

Yuanchong, X. 2005. *Selected Poems and Pictures of the Song Dynasty*. China Intercontinental Press.

Yun, S. 2017. Sou Yun - Public Poetry Database. `https://sou-yun.com/`. [Online; accessed 11-January-2018].

Zhang, X., and Lapata, M. 2014. Chinese poetry generation with recurrent neural networks. In *EMNLP*, 670–680.

Zhang, J.; Feng, Y.; Wang, D.; Wang, Y.; Abel, A.; Zhang, S.; and Zhang, A. 2017. Flexible and creative chinese poetry generation using neural memory. *arXiv preprint arXiv:1705.03773*.