Composing Ci with Reinforced Non-autoregressive Text Generation

Yan Song
University of Science and Technology of China
clk.song@gmail.com

Abstract
Composing Ci (also widely known as Song Ci), a special type of classical Chinese poetry, requires to follow particular format once their tune patterns are given. To automatically generate a well-formed Ci, text generation systems should strictly take into account predefined rigid formats (e.g., length and rhyme). Yet, most existing approaches regard Ci generation as a conventional sequence-to-sequence task and use autoregressive models, while it is challenging for such models to properly handle the constraints (according to tune patterns) of Ci during the generation process. Moreover, consider that with the format prepared, Ci generation can be operated by an efficient synchronous process, where autoregressive models are limited in doing so since they follow the character-by-character generation protocol. Therefore, in this paper, we propose to compose Ci through a non-autoregressive approach, which not only ensure that the generation process accommodates tune patterns by controlling the rhythm and essential meaning of each sentence, but also allow the model to perform synchronous generation. In addition, we further improve our approach by applying reinforcement learning to the generation process with the rigid constraints of Ci as well as the diversity in content serving as rewards, so as to further maintain the format and content requirement. Experiments on a collected Ci dataset confirm that our proposed approach outperforms strong baselines and previous studies in terms of both automatic evaluation metrics and human judgements.

1 Introduction
Ci is a special type of Chinese poetry and different from general poems, where their shape (including lengths and tunes) is determined by tune patterns (i.e., 词牌, also known as brand names) defining the particular syllable and rhythm schemes that a Ci should stick to. Over the past one thousand years, composing Ci has long been an interesting game and continued to present days for Chinese people to demonstrate one’s literary and artistic accomplishments. Given that Ci composition is challenging, to perform this task with artificial intelligence is meaningful since it is a good test for controlled natural language generation under specific constraints (i.e., the length and rhyme of Ci should follow the constraints given by the tune pattern).

1 Related code and resources of the paper are available at https://github.com/synlp/CiGen.

Figure 1: The first half of an example Ci following the tune pattern "念奴娇" (Charm of a Singer), where all essential words in this Ci are highlighted in blue. Translations for each clause are provided for reference.

<table>
<thead>
<tr>
<th>Tune Pattern</th>
<th>Ci Poem</th>
</tr>
</thead>
<tbody>
<tr>
<td>念奴娇</td>
<td>Charm of a Singer</td>
</tr>
<tr>
<td>大江东去,</td>
<td>East flows the mighty river;</td>
</tr>
<tr>
<td>浪淘尽,千古风流人物。</td>
<td>Sweeping away the heroes of times past;</td>
</tr>
<tr>
<td>故垒西边,</td>
<td>This ancient rampart on its western shore</td>
</tr>
<tr>
<td>人道是:三国周郎赤壁。</td>
<td>Is Zhou Yu's Red Cliff of Three Kingdoms's fame;</td>
</tr>
<tr>
<td>乱石穿空,</td>
<td>Huge waves tear banks apart, and foam piles up a thousand drifts of snow;</td>
</tr>
<tr>
<td>惊涛拍岸,卷起千堆雪。</td>
<td>A scene fair as a painting,</td>
</tr>
<tr>
<td>一时多少豪杰。</td>
<td>One time how many heroes!</td>
</tr>
</tbody>
</table>

Countless the brave men here in time gone by!

2 Originally, Ci emerged during the Tang Dynasty (618-907 A.D.), in response to the popularity of foreign musical tunes imported from the Inner Asia, and started its prevailing and was written all through the age in the Song Dynasty (960-1279 A.D.). Overall, there are more than 800 tune patterns known.

3 “念奴” is the name of a famous singer in Tang Dynasty.

4 This translation is from Mr. Xianyi Yang and Mrs. Gladys Yang, famous translators of Chinese and Western literature.
Figure 2: The overall architecture of our CI\textsc{GEN} with non-autoregressive Ci generation guided by essential words and enhanced by reinforcement learning (RL) with a given tune pattern $Z$. The format information is obtained from the tune pattern; \textit{ESS} word generator and Ci generator are used to compute the hidden vectors, namely, $h^t_k$ and $h^t_O$, for generating essential (\textit{ESS}) word (the red block) and candidate Ci (the yellow block), respectively; the generated \textit{ESS} words and candidate Ci are combined to obtain the final Ci (the blue block), which is then used to compute the rewards (including format rewards, rhyme rewards, and diversity rewards) in RL. The dashed red arrows illustrate the rewards sent back from RL. The dashed blue arrow from the format representation to the Ci generator is used to illustrate the vanilla CI\textsc{GEN}, which only generates candidate Ci without the help of \textit{ESS} words.

Recently, text generation models based on deep neural networks (e.g., LSTM (Hochreiter and Schmidhuber, 1997) and Transformer (Vaswani et al., 2017)) have been widely used for poem and Ci generation (Wang et al., 2016a; Yang et al., 2018a; Yi et al., 2018; Yeh et al., 2019; Li et al., 2020; Wu et al., 2021) and demonstrated their validity in doing so. Among previous studies, most mainly follow the conventional text generation paradigm which performs an autoregressive generation process by generating a poem or Ci in a character-by-character manner. However, since Ci is usually longer than classical poem, these autoregressive models face the challenge of losing semantic coherence in-between the beginning and end of a Ci when it is too long and thus might lead to inferior results because a high-quality Ci always requires good coherence in the topic. To illustrate, Figure 1 shows a well-known Ci with the tune pattern “念奴娇” (Charm of a Singer), where every sentence in this Ci are correlated to each other and stick to the main topic. Therefore, Ci shows the following two characteristics which are required to be addressed carefully during its generation process: (1) once the tune pattern is given, the rigid format of the Ci (i.e., the length and rhyme) is determined; (2) different parts in a Ci should show high relevance in semantics (e.g., stick to a particular topic). These characteristics suggest that non-autoregressive models (whose effectiveness for text generation has already been demonstrated in machine translation (Gu et al., 2018), image captioning (Lee et al., 2018), and summarization (Qi et al., 2021)) have their potential to be appropriate choices for this task. Moreover, another advantage of non-autoregressive models is that they are able to generate different parts of a Ci synchronously, which is more efficient compared with autoregressive ones. Still, although non-autoregressive models show aforementioned superiority, there are further improvements needed for Ci composition, such as guiding the model to generate clauses strictly following the length and rhyme requirements of the tune pattern and providing more diversified generation results, which are normally hard to be controlled through conventional supervised/unsupervised functions. Consider that reinforcement learning (RL) is able to guide the parameter optimization process of a model through object-oriented rewards and it has been demonstrated to be effective in many natural language generation tasks such as dialogue generation (Li et al., 2016), paraphrase generation (Li et al., 2017), and image captioning (Qin and Song, 2022), it is expected to be also effective in helping the non-autoregressive models to generate Ci that better
follows the constraints of the tune pattern.
In this paper, we propose, CIGEN, a non-autoregressive model for Ci generation with a given tune pattern, where we employ a key word guided generation process to firstly generate essential (ESS) words (e.g., the ones highlighted in blue in Figure 1) that convey the important meaning for each part (e.g., sentence) and then generate the final full Ci. To further enhance the non-autoregressive model, we apply RL to Ci composition so as to accommodate the formats and rhyming constraints, which are generally hard to learn by the conventional supervised or unsupervised learning methods since it is not easy to design normal loss functions for them. In evaluation, we test our approach on a collected Song Ci dataset\(^3\), where our approach outperforms strong baselines and previous studies on both automatic and human evaluation metrics.

2 The Proposed Approach

Figure 2 illustrates the overall architecture of our CIGEN for Ci composition with the given tune pattern \(Z\), where the format (denoted as \(X = x_1, \ldots, x_t, \ldots, x_T\) with \(x_t\) presenting the format of the \(t\)-th character and \(T\) the number of characters) of Ci is obtained based on the tune pattern \(Z\) and then used to generate the intermediate ESS words (denoted as \(\hat{K}\)), and the candidate Ci (denoted as \(\hat{\mathcal{V}} = \hat{v}_1, \ldots, \hat{v}_t, \ldots, \hat{v}_T\)), then the generated ESS words and the candidate Ci are then combined to obtain the final Ci (denoted as \(\hat{Y} = \hat{y}_1, \ldots, \hat{y}_t, \ldots, \hat{y}_T\)). Therefore, the process of the proposed non-autoregressive approach for Ci composition is formally expressed by

\[
\hat{Y} = C(\hat{V}, \hat{K}) \tag{1}
\]

with

\[
\begin{align*}
\hat{V} &= f(X, \hat{K}) \\
\hat{K} &= f(X) \\
X &= F(Z)
\end{align*} \tag{2}
\]

where \(C\) refers to the combination of candidate Ci and ESS words, \(f\) denotes a general text generation process with the given input, \(F\) extracts the format of the Ci based on the given tune pattern. In the following text, we first illustrate the process to obtain the format representation from the tune pattern, then present the non-autoregressive model for guiding word driven Ci generation, and finally how we use RL enhancement to compose high-quality Ci.

\(^3\)https://github.com/lipiji/SongNet

2.1 Format Representations

One characteristic of Ci is that its format is determined by the tune pattern. To represent the format information, we refer to a previous study (Li et al., 2020) and use the combination of four types of symbols to represent the format \(x_t = (r_t, p_t, s_t, g_t)\), where \(r_t, p_t, s_t,\) and \(g_t\) denote the rhyme (RHY), intra-position (INP), clause-index (CLI), and global-position (GLO) symbols, respectively. Table 1 presents the values of different symbols for the first three clauses (i.e., “大江东去，浪淘尽，千古风流人物。”) in the example in Table 1 for better illustration, and we elaborates the details of these symbols in the following texts.

Rhyme symbols  
Rhyme symbols are designed to illustrate whether the associated characters are required to follow the rhyme of the tune pattern. Specifically, the rhyme symbol \(r_t\) for \(x_t\) has three choices, namely, \(P\) (punctuation), \(R\) (rhyme), and \(O\) (other cases): \(r_t = P\) if \(x_t\) should be a punctuation; \(r_t = R\) if \(x_t\) should follow the rhyme, in which case \(x_{t+1}\) is a punctuation (for Ci, the character that directly precedes the punctuation has to follow the rhyme); \(r_t = O\) otherwise.

Intra-position symbols  
Intra-position symbols \(p_t\) are used to represent the distance of \(x_t\) to the nearest following punctuation. That is, we define this symbol by measuring how far the next punctuation (denoted as \(x_{t'}\)) is to \(x_t\) (where \(t \leq t'\)), and set its value to \(p_t\) to \(b_{t'-t}\). Therefore, \(b_0\) always denotes the punctuation, which enables our model to correctly recognize the boundary of clauses.

Clause-index symbols  
Local-position symbols are used to represent each character that the index of a clause it belongs to. Therefore, \(s_t = c_j\) if the \(t\)-th character is in the \(j\)-th clause in a Ci.

Global-position symbols  
Global-position symbols \(g_t\) are designed to represent the global positional information for each character \(x_t\) and they are demonstrated to be powerful in many previous studies for text generation (Radford et al., 2019; Deng et al., 2020; Lewis et al., 2020; Raffel et al., 2020). In our approach, the global-position symbol \(g_t\) for the \(t\)-th character is \(t\), i.e., \(g_t = t\).

Once all symbol values are obtained for \(x_t\), we map them to their corresponding embeddings, namely, rhyme embedding \(e^r_t\), intra-position embedding \(e^p_t\), clause-index embedding \(e^c_t\), and global-position embedding \(e^g_t\), where we follow the
position embedding mechanism in Transformer (Vaswani et al., 2017) to compute our GLO embeddings. Afterwards, we directly concatenate $\oplus$ the four types of embeddings and obtain the format embedding $e_t^x$ for the $t$-th character by
\[
e_t^x = e_t^e \oplus e_t^p \oplus e_t^g \oplus e_t^n
\]
(3)
To summarize, since the four types of symbols reflect the characteristics of Ci from different aspects, the combination of them (i.e., the format representation $e_t^x$) contains informative features and constraints given by the tune pattern and thus could be used to enhance a model for Ci composition.

### 2.2 Ci Composition with Essential Words

Although non-autoregressive models with aforementioned format representations are able to leverage the format constraints in composing Ci, it is still hard for them to automatically maintain semantic consistency. Consider that the overall emotional tone and topic of a Ci are generally carried by its essential words, we propose to enhance non-autoregressive models through a guided generation process with ESS words. Specifically, the model firstly generates the ESS words with the given tune pattern (i.e., the format representations) and then uses the generated ESS words to guide the rest generation process. Therefore, our model is able to learn the potential relation between the format and the overall emotion tone carried by the ESS words and leverage them for the later Ci composition.6

For ESS words generation, our model applies an encoder (denoted as $f_1$) to the format representation $E^x = e_1^x, \cdots, e_T^x$ and obtain a sequence of hidden vectors $H^k = h_1^k, \cdots, h_T^k$ by
\[
H^k = f_1(E^x)
\]
(4)
It is worth noting that $f_1$ takes the matrix $E^x$ and computes the matrix $H^k$ through a single forward pass, which differs from conventional autoregressive approach that generates a single vector step by step. Then the hidden vector $h_t^k$ is then fed into a fully connected layer with the $\text{softmax}$ classifier to predict the ESS character $\hat{k}_t$ for the input $x_t$:
\[
\hat{k}_t = \text{softmax}(W_1 \cdot h_t^k + b_1)
\]
(5)
where $W_1$ and $b_1$ are the trainable matrix and bias vector in the fully connected layer, respectively.

With ESS words, for Ci generation, we firstly map all generated ESS characters $\hat{k}_t$ to their embeddings $e_t^k$ and then add the format representation $e_t^x$ to the resulting embeddings through
\[
h_t^i = e_t^i + e_t^k
\]
(6)
Afterwards, similar to the generation process of ESS words, we use another encoder (which is denoted as $f_2$) and computes the output matrix via a process similar to $f_1$) to process the obtained $H^i = h_1^i, \cdots, h_T^i$ and obtain the hidden vectors $H^O = h_1^O, \cdots, h_T^O$ via
\[
H^O = f_2(H^i)
\]
(7)
where $h_t^O$ is fed into a fully connected layer with the $\text{softmax}$ classifier to predict the character $\hat{v}_t$ for each $x_t$ in the candidate Ci:
\[
\hat{v}_t = \text{softmax}(W_2 \cdot h_t^O + b_2)
\]
(8)
where $W_2$ and $b_2$ are the trainable matrix and bias vector. Finally, to take the advantage of the generated ESS words, we combine the ESS words and the candidate Ci based on the following rule:
\[
\hat{y}_t = \begin{cases} 
\hat{v}_t & \text{if } \hat{k}_t = [N] \\
\hat{k}_t & \text{otherwise} 
\end{cases}
\]
(9)
so as to obtain the final resulted Ci, $\hat{Y}$.

Different from the conventional autoregressive

---

6Actually, some particular formats and rhymes are frequently used to express certain types of emotions.
text generation approaches, for both ESS word and Ci generation, our model generates all characters synchronously, which allows our model to efficiently leverage context information in a one-time encoding and decoding process in Ci composition.

In the training process, we compare the generated ESS words and the final resulted Ci with the ground truth and compute the loss $L_K$ and $L_Y$ for them, respectively, which are further used to update the model parameters through backpropagation.

### 2.3 Enhancement with RL

Although using the non-autoregressive model conditioning on format and rhyme is able to generate more satisfying Ci than models without such restriction (e.g., systems designed for poetry generation), there are still gaps between the performance of automatically generated Ci and those composed by poets. To address this problem, we propose to incorporate reinforcement learning (RL) into our non-autoregressive model so as to further improve the quality of Ci composition. In doing so, we regard the entire generation process as a two-state reward maximization task. Therefore, in training each instance, the agent (i.e., the model) starts from the initial state $S_0$, which is the input format, then selects an action (i.e., generated Ci $\hat{Y}$) according to the policy (i.e., $\pi(\theta)$) with $\theta$ denoting all model parameters, and receives a reward $r$ and arrives at the terminal state. Specifically, the total reward is a linear combination of the format, rhyme, and diversity scores via

$$r = \lambda_1 r_f + \lambda_2 r_r + \lambda_3 r_{d_1} + \lambda_4 r_{d_2}$$

(10)

where $\lambda_i$ ($i \in \{1, 2, 3, 4\}$) are hyper-parameters; $r_f$ is the score for format, which is the number of correctly segmented sentences/lines in $\hat{Y}$; $r_r$ is the score for rhyme which is the number of predicted characters that correctly follow the rhyme requirement of the given tune pattern; $r_{d_1}$ and $r_{d_2}$ are the scores for uni-gram and bi-gram diversities, respectively, which are the number of unique uni-grams and bi-grams in the generated Ci.

To solve the reward maximization problem, we follow the REINFORCE algorithm (Williams, 1992) with loss and corresponding gradient

$$L_{RL}(\theta) = -\mathbb{E}_{\pi_\theta}(r) = -\sum_\gamma p(\gamma; \theta) \cdot r(\gamma)$$

(11)

and

$$\nabla_{\theta} L_{RL}(\theta) = -\sum_\gamma p(\gamma; \theta) \cdot r(\gamma)$$

(12)

respectively. The gradient is estimated by a single Monte-Carlo sampling $\hat{Y} = \{\hat{y}_1, ..., \hat{y}_T\}$ through

$$\nabla L_{RL}(\theta) \approx -\nabla_{\theta} \log p(\hat{Y}; \theta) r(\hat{Y})$$

(13)

However, the estimation of gradient is of high variance. Therefore, we follow Rennie et al. (2017) and introduce a baseline function that is independent with the action $\hat{Y}$. Therefore, the refined loss and gradient estimations are formalized as

$$L_{RL}(\theta) = -\mathbb{E}_{\pi_\theta}(r(\hat{Y}) - r(\hat{Y}'))$$

(14)

and

$$\nabla L_{RL}(\theta) \approx -\nabla_{\theta} \log p(\hat{Y}; \theta) (r(\hat{Y}) - r(\hat{Y}'))$$

(15)

respectively, where $\hat{Y}'$ denotes the generated Ci selected using top-$k$ sampling.

As a result, the overall training loss is formalized as a linear combination of all losses from the aforementioned steps, including $L_K$, $L_Y$, and $L_{RL}$.

$$L = \alpha L_K + \beta L_Y + \gamma L_{RL}$$

(16)

where $\alpha$, $\beta$, and $\gamma$ are hyper-parameters to control the effect of $L_K$, $L_Y$, and $L_{RL}$, respectively.

### 3 Experiment Settings

#### 3.1 Dataset

To evaluate the performance of our approach, we run experiments with Song Ci dataset\(^9\), where there is no official train/dev/test split for this dataset, so that we randomly split the data into training, development, and test sets, with the statistics reported in Table 2. Since no ESS words’ annotations are provided in the original dataset, we automatically annotate ESS words and regard them as the ground truth in training our model. In doing so,

\(\text{Table 2: The statistics of our experiment dataset in terms of the number of Ci, characters, and ESS words in the training, development, and test set, respectively.}\)

<table>
<thead>
<tr>
<th></th>
<th>Ci #</th>
<th>Character #</th>
<th>ESS Word #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>17,733</td>
<td>1.2M</td>
<td>112.5K</td>
</tr>
<tr>
<td>Dev</td>
<td>1,071</td>
<td>67.5K</td>
<td>6.5K</td>
</tr>
<tr>
<td>Test</td>
<td>1,169</td>
<td>82.6K</td>
<td>7.5K</td>
</tr>
</tbody>
</table>

\(^9\text{https://github.com/lipiji/SongNet.}\)
we randomly sample 1,000 Ci from the dataset and invite two annotators to manually mark ESS words that convey important meaning in each Ci. Next, we use the annotated Ci as training data to train a BERT-based (Devlin et al., 2019) ESS word annotator (which is similar to a named entity annotator) following the sequence labeling paradigm. Then, we apply the trained annotator to the entire dataset and obtain the “ground truth” ESS words. The statistics of the auto-annotated ESS words in the train/dev/test sets are also reported in Table 2.

### 3.2 Implementation Details

Since the quality of text representation plays an important role in many natural language processing tasks (Han et al., 2018; Radford et al., 2019; Tian et al., 2020; Lewis et al., 2020; Diao et al., 2020; Raffel et al., 2020), we use the well-performed Transformer (Vaswani et al., 2017) architecture for both ESS word and Ci generation. Specifically, for both Transformer encoders (i.e., $f_1$ and $f_2$), we use 6 layers of multi-head attentions, with 12 heads and the dimension of the hidden vectors set to 768. For the $\lambda$ in RL rewards, we treat the rewards from format, rhyme, and diversity equally with $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 0.25$ in Eq. (10). Besides, we use $k = 32$ for the $top-k$ sampling to compute the refined loss (i.e., see Eq. (11)).

We train all models including the one with RL for 30 epochs over all training data. For evaluation, we follow previous studies (Li et al., 2020) and evaluate model performance from diversity, format, and rhyme using Macro-F1 (MA) and Micro-F1 (MF), where the diversity is evaluated based on the distinctness of uni-grams and bi-grams, respectively; the speed of different models are computed via the number of generated Ci per second.

### 4 Results and Analysis

#### 4.1 Overall Results

Table 3 reports the experimental results of our approach with different settings: CI\textit{GEN} is the model that uses only $f_2$ and treats the candidate Ci as the final one. CI\textit{GEN} + ESS uses the ESS words to guide the Ci generation process, and CI\textit{GEN} + ESS + RL is our full model which leverages both ESS words and RL. The results of our experiment using SongNet (Li et al., 2020) is also reported for comparison. We also present the inference speed (i.e., the number of generated Ci per second) of all models. Overall, there are several observations.

First, compared with CI\textit{GEN} that directly generates Ci, the model enhanced by ESS word guided generation (i.e., CI\textit{GEN} + ESS) achieves higher performance with respect to all evaluation metrics. This observation indicates that, CI\textit{GEN} + ESS is able to learn from the ESS words that carry important semantic or topic information, and thus allows the model to generate a Ci with more coherent and meaningful expression. On the contrary, CI\textit{GEN} does not benefit from the ESS words so that it leads to inferior performance in all evaluation metrics.

Second, comparing CI\textit{GEN} + ESS and our full model CI\textit{GEN} + ESS + RL, it is observed that the full model with RL further improves the performance of CI\textit{GEN} + ESS on all evaluation metrics, which demonstrates the effectiveness of RL in Ci composition. A possible explanation can be elaborated as follows. With the modeling of format, rhyme, and diversity rewards through RL, the full

---

10We use the BERT-base-Chinese obtained from https://github.com/google-research/bert.

11It can be implemented with pre-trained encoders, e.g., we tried ZEN (Song et al., 2021) and it can obtain better results than training a Transformer from scratch according to human evaluation. One could infer from this setting that both $f_1$ and $f_2$ are interchangeable with other similar models.

12We tried different $k$ values in the experiments and locate that $k = 32$ is optimal and achieves the best performance.
Table 4: Experimental results of our full model CIGEN + ESS + RL, with one of the four types of rewards (i.e., format, rhyme, uni-gram and bi-gram diversity) ablated. E.g., “- Format ($r_f$)” means that format reward is ablated.

<table>
<thead>
<tr>
<th>Models</th>
<th>Diversity</th>
<th>Format</th>
<th>Rhyme</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MA-U</td>
<td>MI-U</td>
<td>MA-B</td>
</tr>
<tr>
<td>CIGEN + ESS + RL</td>
<td>74.36</td>
<td>3.13</td>
<td>98.73</td>
</tr>
<tr>
<td>– Format ($r_f$)</td>
<td>73.05</td>
<td>2.31</td>
<td>97.77</td>
</tr>
<tr>
<td>– Rhyme ($r_r$)</td>
<td>73.91</td>
<td>2.20</td>
<td>97.86</td>
</tr>
<tr>
<td>– Uni-gram diversity ($r_{d_1}$)</td>
<td>71.71</td>
<td>2.14</td>
<td>97.65</td>
</tr>
<tr>
<td>– Bi-gram diversity ($r_{d_2}$)</td>
<td>71.73</td>
<td>2.16</td>
<td>97.43</td>
</tr>
</tbody>
</table>

Table 5: Human evaluation results from different models with respect to four metrics, where higher scores (whose range is [1, 3]) refer to higher qualities. “Con.,” “Flu.,” “Mea.,” and “Poe.” denote the scores for consistency, fluency, meaning, and poeticness, respectively, and “Avg.” reports the average of them.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CIGEN</td>
<td>2.30</td>
<td>1.55</td>
<td>1.55</td>
<td>1.80</td>
<td>1.80</td>
</tr>
<tr>
<td>CIGEN + ESS</td>
<td>2.75</td>
<td>1.75</td>
<td>1.90</td>
<td>2.10</td>
<td>2.13</td>
</tr>
<tr>
<td>CIGEN + ESS + RL</td>
<td>2.95</td>
<td>1.90</td>
<td>2.20</td>
<td>2.30</td>
<td>2.34</td>
</tr>
</tbody>
</table>

model is able to learn to force the generation process to satisfy the restrictions (e.g., format and rhyme constraints) of the given tune pattern and thus achieves higher performance than CIGEN + ESS that does not target to such restrictions.

Third, our full model CIGEN + ESS + RL outperforms SongNet (Li et al., 2020) (which uses an autoregressive approach for Ci generation) with respect to all evaluation metrics. This observation not only demonstrates the effectiveness of our proposed approach, but also indicates that non-autoregressive models are also plausible solutions for composing Ci and format-specific text genres. In addition, benefiting from the nature of non-autoregressive approach (i.e., the model is able to generate all characters at the same time), our models are able to generate CIs around 100 times faster than SongNet, where the simplest CIGEN is the fastest one that is able to generate around 43 CI in each second. This comparison demonstrates the superior efficiency of applying synchronous procedure to Ci composition and shows its great potential to be applied to real applications that in similar scenario and require high generation speed.

### 4.2 Effect of Different Rewards

In our full model with RL (i.e., “CIGEN + ESS + RL”), we compute the final reward $r$ for RL by averaging the rewards from format (i.e., $r_f$), rhyme (i.e., $r_r$), uni-gram diversity (i.e., $r_{d_1}$), and bi-gram diversity (i.e., $r_{d_2}$) (see Eq. (10)). To further investigate the effect of the designed rewards, we perform ablation study where one of the four types of reward is ablated. Table 4 reports the experimental results, where the best and the worst result for each evaluation metric are highlighted in boldface and underlines, respectively. There are several observations. First, overall, the ablation of any one of the reward types hurts model performance on all metrics, which demonstrates that all types of rewards contribute to the quality of Ci composition. Second, the ablation of a particular type of reward would strongly hurt the model performance on its corresponding metric. For example, the worst performance on rhyme evaluation metrics is achieved when rhyme reward (i.e., $r_r$) is ablated. So that each reward does confirm its value in helping generate better Ci from different aspects. To summarize, our observations demonstrate the effectiveness of the full model with RL to learn from all types of reward and thus to generate Ci satisfying different types of evaluation metrics.

### 4.3 Human Evaluation

Following previous studies (Li et al., 2018; He et al., 2012; Zhang and Lapata, 2014; Wang et al., 2016b; Yu et al., 2017), we further conduct human evaluation on those CIs generated from different models (i.e. CIGEN, CIGEN + ESS, and CIGEN + ESS + RL), where four different metrics (namely, consistency, fluency, meaning, and poeticness) are considered. To explain, consistency evaluates the theme consistence; fluency measures the grammatical correctness; meaning stands for the meaningfulness of the content; and poeticness exams whether the Ci follows the attributes of poetry. We randomly sample 50 CIs from the test set and invite five human evaluators who are familiar with Chinese poetry to score each CI based on the aforementioned four.
metrics, where the score is one of \{1, 2, 3\} with 1 for poor, 2 for medium, and 3 for good. The evaluation is conducted in a blind review manner, where evaluators are provided with the Ci generated from different models but they cannot locate which model generates the given Ci. We report the scores for all evaluation metrics as well as the overall average score (AVG.) in Table 5. Similar to the observations from Table 3, in Table 5, CI GEN + ESS with essential word guided generation process achieves better performance than CI GEN and CI GEN + ESS + RL further improves CI GEN + ESS with the help of RL. To conclude, human evaluation not only reveals the capability of the proposed CI GEN, but also further confirms the effectiveness of ESS words and RL for CI composition.

### 4.4 Case Study

To qualitatively investigate different models, especially the effect of ESS words and RL, we conduct a case study with an example input tune pattern “捣练子” (Daolianzi Theme). Figure 3 illustrates a reference Ci and the generated ones from three different models (i.e., CI GEN, CI GEN + ESS, and CI GEN + ESS + RL) given the tune pattern, where the rhyme characters required by the tune pattern are underlined in the reference and the generated Ci; the ESS words in the reference (automatically labeled) and in the Ci generated by CI GEN + ESS and CI GEN + ESS + RL are highlighted in blue color. It is observed that CI GEN generates an inferior output that is irregular in terms of rhythm where the underlined rhyming characters (i.e., “西”, “为”, “夜”) of the generated clauses do not follow the same vowels. When ESS words are used, CI GEN + ESS is able to benefit from the ESS words and thus generates most of the ESS words at the expected positions. However, we notice that the Ci generated by CI GEN + ESS is still not perfect: the underlined rhyming character “蜻” and “情” in the first and fourth clause are homophones, which is normally avoided in poem and Ci composition but using characters with different pronunciation while sharing the same rhyme. When RL is applied, compared with CI GEN + ESS, CI GEN + ESS + RL

<table>
<thead>
<tr>
<th>Tune Pattern</th>
<th>Ci</th>
<th>Translation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference</td>
<td>妾练子</td>
<td>Under the falling moon, (a lady) faces the north wind. With thousands of times the pestle hit the clothes (to make the clothes flatten), the plank is about to be broken. (The lady) is awake not for flattening the clothes. Not just tonight, every night is like a year to her (by deadly missing her husband).</td>
</tr>
<tr>
<td>CI GEN</td>
<td>南岳顶，日落西</td>
<td>On the top of the south mountain, (I) see the west sunset. The green peaks from all sides enter the layers of clouds. The creek is flowing; the monkeys are chirping. All the frightened birds in the woods are flying into the sky.</td>
</tr>
<tr>
<td>CI GEN + ESS</td>
<td>春风起，雨还晴</td>
<td>The spring breeze is blowing, the rainy days are clearing. The green plums on the tree comes into view. Please do not say there is nowhere to look for signs of spring. The falling flowers and catkins always deliver silent affection.</td>
</tr>
<tr>
<td>CI GEN + ESS + RL</td>
<td>我依于千山不能眠，一帘疏影成仙</td>
<td>(I am standing) at the foot of a mountain, by the stream, (I) see the flying flowers in the entire garden which reminds me of the passing years. (I) rely on the railing alone and cannot fall asleep. Seeing a sparse scene of shadows, I feel like I am about to become a fairy.</td>
</tr>
</tbody>
</table>

Figure 3: The comparison of a reference and the generated Ci from different models with a given tune pattern, where the English translation is also provided for better understanding. ESS words are highlighted in blue color and the rhyming characters required by the tune pattern are underlined in the reference and all generated Ci.
is able to generate more coherence and consistent content, having better artistic conception and diversified in generating rhyming characters as well as the ESS words, with their positions matching the ones in the reference. generate high-quality Ci.

5 Related Work

Chinese Ci generation is generally considered as one type of Chinese archaic style text generation tasks. In addition to Ci generation, such tasks also include couplet and classical poetry generation, where they have different concerns regarding to particular restrictions. Specifically, couplet generation (Jiang and Zhou, 2008; Zhang et al., 2018; Fan et al., 2019; Gao et al., 2021; Song, 2022) is a strictly conditioned text generation task where the generated text (subsequent clause) has to correspond to the input text (antecedent clause) in almost all aspects, such as rhyme, length, syntactic and semantic correspondence, etc. Classical poetry generation (Zhang and Lapata, 2014; Zhang et al., 2017; Li et al., 2018; Yang et al., 2018b, 2019; Chen et al., 2019; Deng et al., 2020) normally focuses on unconditioned text generation with limited format constraints, where there are typically two poem types, i.e., five-character and seven-character quatrain. Different from the two tasks, Ci generation is more flexible than couplet but less than poem, in terms of using tune patterns for restriction. To the best of our knowledge, there are 871 different types of tune patterns with each having its own format requirements. In performing Ci generation, studies are much less than that for couplet and poem generation, recent ones (Wang et al., 2016a; Li et al., 2020; Luo et al., 2021) leverage deep learning based models and achieve outstanding performance, where most of them regard the task as a conventional sequence-to-sequence task and use autoregressive approaches. To further improve the task, there are studies applying enhanced modules such as attentions (Wang et al., 2016a) and pre-training language models (Li et al., 2020).

Compared to previous studies based on deep neural networks, this work takes the advantage of the properties of Ci (i.e., the format is determined once the tune pattern is given and different parts of Ci should stick to a particular topic) and provides an alternative solution for Ci generation through a non-autoregressive method, which allows our model to generate Ci efficiently. Particularly, the generation process guided by essential words and RL with carefully designed rewards further facilitate the explicit accommodation of the rigid constraints for Ci, leading to better results in all evaluations.

6 Conclusion

In this paper, we propose a non-autoregressive model named CiGEN for Ci composition, which is further enhanced by an essential word guided generation process and RL. Specifically, our model firstly generates the ESS words that convey important meaning or topic of Ci and then use these ESS words to complement the generation of the entire Ci. In addition, we design a set of RL rewards based on format, rhyme, and diversity (including uni-grams and bi-grams based measures) to enhance the model by further accommodating the constraints from the tune pattern, for the purpose of solving the problem that normal loss functions for conventional supervised/unsupervised methods cannot be applied to such constraints. Experimental results and analyses on a Song Ci dataset confirm the validity of our proposed model, with its evaluation outperforms strong baselines and previous studies with respect to different evaluation metrics. Moreover, owing to the non-autoregressive characteristic, the inference speed of our model also shows its great superiority over the autoregressive ones. Therefore, the effectiveness and efficiency indicates that our model design has its potential to be implemented to similar text generation scenarios.

References


