ChipSong: A Controllable Lyric Generation System for Chinese Popular Song

Nayu Liu[1], **Wenjing Han**[1], **Guangcan Liu**[1], **Peng Zhou**[1], **Ran Zhang**[1], **Xiaorui Wang**[1], **Huabin Ruan**[2*]
[1] Kuaishou Technology Co., Ltd, Beijing, China
[2]School of Life Sciences, Tsinghua University, Beijing, China
{liunayu, hanwenjing, liuguangcan, zhoupeng, zhangran, wangxiaorui}@kuaishou.com, ruanhuabin@tsinghua.edu.cn

Abstract

In this work, we take a further step towards satisfying practical demands in Chinese lyric generation from musical short-video creators, in respect of the challenges on songs' format constraints, creating specific lyrics from open-ended inspiration inputs, and language rhyme grace. One representative detail in these demands is to control lyric format at word level, that is, for Chinese songs, creators even expect fix-length words on certain positions in a lyric to match a special melody, while previous methods lack such ability. Although recent lyric generation community has made gratifying progress, most methods are not comprehensive enough to simultaneously meet these demands. As a result, we propose *ChipSong*, which is an assisted lyric generation system built based on a Transformer-based autoregressive language model architecture, and generates controlled lyric paragraphs fit for musical short-video display purpose, by designing 1) a novel *Begin-Internal-End* (BIE) word-granularity embedding sequence with its guided attention mechanism for word-level length format control, and an explicit symbol set for sentence-level length format control; 2) an open-ended trigger word mechanism to guide specific lyric contents generation; 3) a paradigm of *reverse order training and shielding decoding* for rhyme control. Extensive experiments show that our ChipSong generates fluent lyrics, with assuring the high consistency to pre-determined control conditions.

1 Introduction

Lyric generation is a recent emerging topic in intelligent music research community, which has attracted increasing attention and gained progress in the past few years (Watanabe et al., 2018; Manjavacas et al., 2019; Fan et al., 2019; Li et al., 2020; Zhang et al., 2020a; Nikolov et al., 2020; Sheng et al., 2021). Meanwhile, observing a large amount of music lovers, amateurs, and professional musicians are gathering on today's fast growing Chinese short-video platforms (e.g., Kwai, TikTok, Wesee, etc.), where they create and post musical short-videos actively, with purpose to obtain more *Follow*s and *Like*s from general population; we believe it is worth to customize a lyric generation system for their short-video display purpose.

Hence, in this paper, we aim to put more emphasis on assisting creators from practical short-video scenario with realistic demands. In oder to collect their real demands, a qualitative investigation with 85 potential users (ages: 18∼40 years; 42 female, 43 males; 19 full-time musicians, 66 part-time musicians) is conducted at the very first stage. Here, we briefly release 4 representative demands as follow: 1) *short lyric paragraphs* are required to fit in short-video durations, mostly under 60 sec. (Zhang et al., 2020b); 2) *open-ended inspiration inputs* are desired to guide specific content generation from various creators; 3) *length format controlling at* sentence and even *word level* is expected to strictly match melody length format for flexible creation intents, where a Chinese word is generally composed of multiple characters (e.g., "爱" means *love*, "爱好" means *hobby*, "爱尔兰" means *Ireland*) and one character sounds one syllable; 4) *rich rhyme patterns* are needed for smooth song singing. Although recent progress has been made on lyric generation, previous works are not comprehensive enough to simultaneously meet these customized demands; What's more, as far as we know, none of the existing work supports word-level length format control.

Taking the above challenges in mind, we develop *ChipSong*, a lyric generation system, to assist musical short-video creators for **Chi**nese **p**opular **song** creation. As shown in Figure 1, with ChipSong, a creator is encouraged to input a group of open-ended words (which are referred to as *trigger words* in the following) to represent his/her inspiration,

[*]Corresponding author

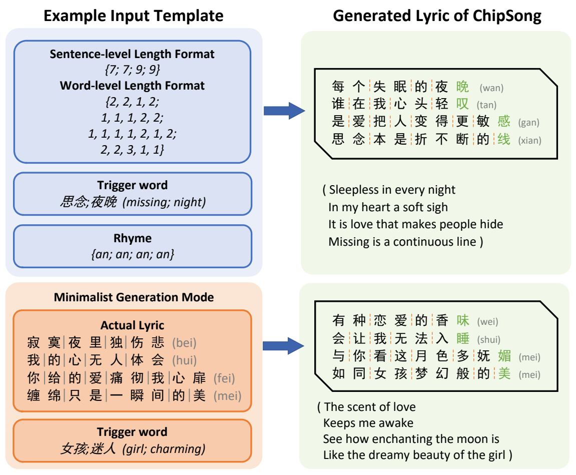
Example Input Template
Generated Lyric of ChipSong
Sentence-level Length Format
{7; 7; 9; 9}
Word-level Length Format
{2, 2, 1, 2;
1, 1, 1, 2, 2;
1, 1, 1, 1, 2, 1, 2;
2, 2, 3, 1, 1}
Trigger word
思念;夜晚 (missing; night)
Rhyme
{an; an; an; an}
每 个 失 眠 的 夜 晚 (wan)
谁 在 我 心 头 轻 叹 (tan)
是 爱 把 人 变 得 更 敏 感 (gan)
思 念 本 是 折 不 断 的 线 (xian)
(Sleepless in every night
In my heart a soft sigh
It is love that makes people hide
Missing is a continuous line)
Minimalist Generation Mode
Actual Lyric
寂 寞 夜 里 独 伤 悲 (bei)
我 的 心 无 人 体 会 (hui)
你 给 的 爱 痛 彻 我 心 扉 (fei)
缠 绵 只 是 一 瞬 间 的 美 (mei)
Trigger word
女孩;迷人 (girl; charming)
有 种 恋 爱 的 香 味 (wei)
会 让 我 无 法 入 睡 (shui)
与 你 看 这 月 色 多 妩 媚 (mei)
如 同 女 孩 梦 幻 般 的 美 (mei)
(The scent of love
Keeps me awake
See how enchanting the moon is
Like the dreamy beauty of the girl)

Figure 1: Our ChipSong system generates lyrics based on the preset template including length formats, trigger words, and rhymes. The blue box shows an example template, and the green box shows the generated lyrics. A minimalist generation mode of ChipSong is shown in the orange box, which extracts the format and rhyme of the actual lyric to imitatively generate a lyric. English translations of Chinese are provided in parentheses.

and a sequence of numbers to tell the length of each lyric line or even each word (a combination of Chinese characters) in a line for matching melody length. The creator can also choose rhyme for the last character in each line from Chinese 14-rhyme[1] groups. Moreover, a minimalist generation mode is provided, where the creator only has to input trigger words and an actual lyric he/she is interested in, then ChipSong will extract the lyric's format and rhyme pattern, and generate a new lyric according to the input trigger words and the extracted format and rhyme, thus fully imitating the original lyric for making a cover song version.

To ensure the relevance of generated lyrics with the above controlling attributes, following efforts are made in this paper: 1) A large corpus of 848K Chinese lyrics are gathered, and tailored according to proper lengths for short-video display. 2) A two-stage sampling strategy is designed to produce a large number of potential trigger words from lyrics themselves without human annotation, and an autoregressive language model is self-supervisedly trained to complete the whole lyric sequence according to partially-observed trigger words, thus stimulating users' open-ended inspiration inputs. 3) Both explicit and implicit control methods are proposed to arrange the format of sentence- and word- level length respectively, where sentence length is controlled via explicit character sets, and word length is controlled via a well-designed implicit *Begin-Internal-End* (BIE) word-granularity embedding sequence with its guided attention mechanism. 4) A strategy of *reverse order training & shielding decoding* is designed to learn a reverse language model, guaranteeing fluent text generation following rhyme control, inspired by the observation that, during lyric creation, humans usually first determine which word to use in the rhyming position of a sentence and then create the rest of that sentence based on the rhyming word. Experimentally, both automatic and human evaluations demonstrate that our ChipSong system generates fluent lyrics with high consistency to pre-determined control conditions.

In summary, oriented to the actual demands of musical short-video creators, we develop ChipSong, a controllable lyric generation system, which can achieve fine-grained control over lyric generation by the proposed control methods for trigger words, format and rhyme. Especially, to the best of our knowledge, ChipSong is the first lyric generation system that can precisely control the word-level length format.

2 Related Work

Recent lyric generation works can be broadly categorized into three groups according to their cared artistic genres: 1) *hip-pop* generation, creating hip-pop lyrics with distinctive rhymes and rhythms constrains (Manjavacas et al., 2019; Nikolov et al., 2020; Xue et al., 2021); 2) *poetry* generation, creating some special text paradigms, such as Shakespeare's Sonnet (Oliveira et al., 2017; Li et al., 2020), Chinese Classical Poetry (Guo et al., 2019; Hu and Sun, 2020; Li et al., 2020), and Chinese Couplet (Yan et al., 2016), etc; 3) *popular song* generation, creating full-text lyrics (Watanabe et al., 2018; Zhu et al., 2018; Lee et al., 2019; Fan et al., 2019; Zhang et al., 2020a; Sheng et al., 2021) or polishing draft lyrics (Zhang et al., 2020a) for popular songs. Our ChipSong is actually a lyric generation system within the third group, and especially for Chinese popular songs. Moreover, different from previous lyric generation works, which were mostly *model*-oriented for natural paragraphs generation and excluded explicit user profiles from practical application scenarios, ChipSong customizes functions to generate lyrics for users from practical short-video scenario with re-

[1]About Chinese 14-rhyme

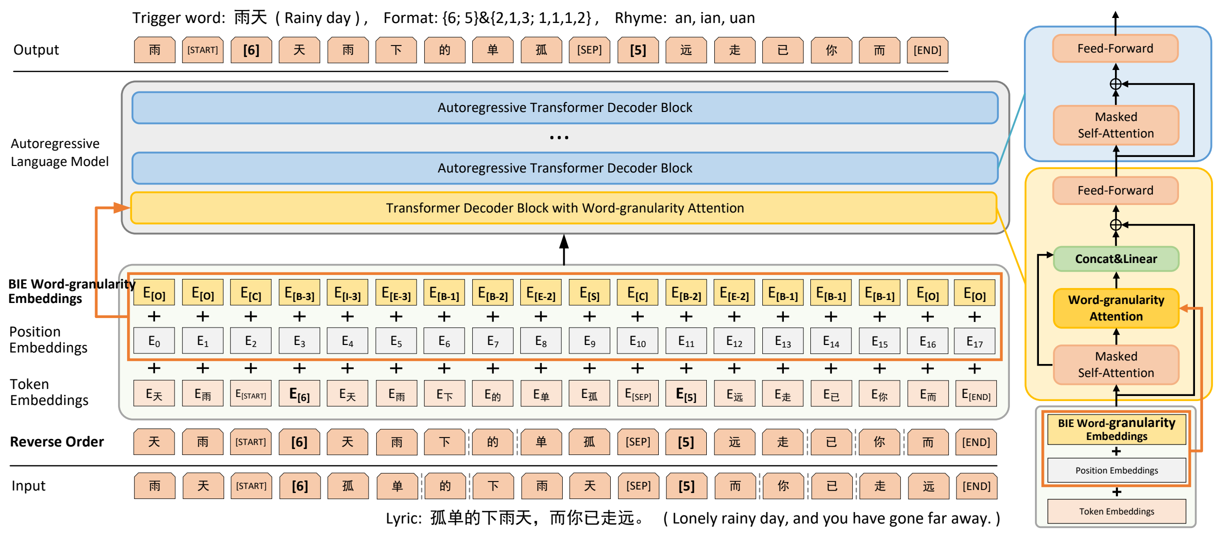
Trigger word: 雨天（Rainy day）, Format: {6; 5}&{2,1,3; 1,1,1,2}, Rhyme: an, ian, uan
Output
雨 [START] [6] 天 雨 下 的 单 孤 [SEP] [5] 远 走 已 你 而 [END]
Autoregressive Language Model
Autoregressive Transformer Decoder Block
...
Autoregressive Transformer Decoder Block
Transformer Decoder Block with Word-granularity Attention
BIE Word-granularity Embeddings
E[O] E[O] E[C] E[B-3] E[I-3] E[E-3] E[B-1] E[B-2] E[E-2] E[S] E[C] E[B-2] E[E-2] E[B-1] E[B-1] E[B-1] E[O] E[O]
Position Embeddings
E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 E10 E11 E12 E13 E14 E15 E16 E17
Token Embeddings
E天 E雨 E[START] E[6] E天 E雨 E下 E的 E单 E孤 E[SEP] E[5] E远 E走 E已 E你 E而 E[END]
Reverse Order
天 雨 [START] [6] 天 雨 下 的 单 孤 [SEP] [5] 远 走 已 你 而 [END]
Input
雨 天 [START] [6] 孤 单 的 下 雨 天 [SEP] [5] 而 你 已 走 远 [END]
Lyric: 孤单的下雨天，而你已走远。（Lonely rainy day, and you have gone far away.）
Feed-Forward
Masked Self-Attention
Feed-Forward
Concat&Linear
Word-granularity Attention
Masked Self-Attention
BIE Word-granularity Embeddings
Position Embeddings
Token Embeddings

Figure 2: The left figure shows the overall architecture of ChipSong. The right figure shows the internal structure of layers; for simplicity, we've omitted the drawing of residual connection and layer normalization.

alistic demands regarding length format, trigger word, and rhyme, simultaneously.

Furthermore, detailed comparison between previous lyric generation works and ChipSong are conducted as follows, from implementation view. First, when it comes to length format control of lyrics, we only notice works (Shen et al., 2019; Li et al., 2020) with sentence-level length control, and no work currently with word-level length control. Second, most of previous works lacked sufficient abilities to deal with open-ended inputs to guide lyric content (Potash et al., 2015, 2018), as a result of the shortage of annotated training data. Fan et al. (2019) and Lu et al. (2019) regarded the user input as the first sentence and generate a continuation of lyric, but it tends to deviate from the initial input as the continuation progresses. Although, Zhang et al. (2020a) designed an interactive lyric creation system to handle the open-ended inputs, as a demo description work, it did not release sufficient implementation details and experimental evaluations. Third, in consideration of the creator's demand for rhyme control, previous works employed various rhyme modeling methods: Nikolov et al. (2020) selected output words from the a list of candidate rhyming words at the rhyming position, while forcibly adding a rhyming word could result in incoherent text in the rhyming position; SongNet (Li et al., 2020) proposed a rigid format control method to realize the rhyme modeling for Chinese lyrics; The recent DeepRapper (Xue et al., 2021) focused on continuous N-gram rhyme and rhythm modeling for rap generation, while we work on unigram rhyme control for popular songs.

3 Method

3.1 Overview

As shown in Figure 2, a Transformer-based autoregressive language model architecture is adopted as the backbone of ChipSong for lyric generation. And by modifying the internal model structure and utilizing processed external feature inputs, we apply the modeling of control conditions for length format, trigger word, and rhyme. In the subsequent section arrangement, we first describe the control condition inputs for ChipSong, and then describe the proposed condition control methods in detail.

3.2 User Inputs

As shown in Figure 1, the user specifies the conditional templates to formulate the lyric generation, and ChipSong generates the lyrics that meet the corresponding control conditions.

Trigger word: enter a few words that are separated by ";" to render the lyric content.

Format: enter each line length and each word (i.e., a combination of Chinese characters) length of each line in the lyric, where sentences' lengths are separated by ";", and words' lengths are separated by ",". For example, enter "7; 7; 9; 2, 2, 3, 1, 1" means to generate four lines of the lyric with lengths of 7, 7, 9, and 9, respectively, and in the last sentence, the word length arrangement is specified as 2,2,3,1,1. Users can also not specify the full template, and the system retrieves similar templates to complement the length format; or directly input a lyric, and the system extracts the length format for imitative writing.

Rhyme: enter the rhyme of the last word in each sentence. Rhymes are separated by ";". For example, input "ui;ui;ui;ui", the generated lyrics keep the rhyme of the last word of each sentence match with "ui". Users can also not specify rhymes, and the system freely generates sentences.

3.3 Sentence-level Format Control

An explicit character set C_S is designed to control the length of each line of the lyrics, just like "$[CLS]$" and "$[SEP]$" in BERT (Devlin et al., 2018), which is constructed as follows:

$$..., [START], [4], a_1, a_2, a_3, a_4, [SEP], [5], \\ a_1, a_2, a_3, a_4, a_5, [END]$$

where $[SEP]$ is the interline delimiter, a_i is the i-th character of a sentence, $[START]$ and $[END]$ are the beginning and end of a lyric, [4] and [5] represent that the next sentence length is 4 and 5, respectively. We assign 50 learnable character embeddings $\{[1], [2], [3], ..., [50]\}$ to C_S to represent the line length from 1 to 50, which are embedded in the lyric sequence as explicit supervisory information for training. The control character is placed after the sentence separator $[SEP]$ and before the beginning of the sentence to learn the correspondence between the control symbol and the sentence length. During prediction, the format control character entered by the user is inserted after the initial $[START]$ token and the generated $[SEP]$ token to achieve the length control of lines.

3.4 Word-level Format Control

Beyond the sentence-level format control, word-level format control arranges lyrics in a more refined way, benefiting fine-grained lyrics' adjustment or imitative writing lyrics. Unlike sentence-granularity format control, the explicit character control strategy makes input too verbose, and the unidirectional masked self-attention of autoregressive language model cannot model the uninput control symbols, which is difficult to reconcile and arrange the fixed-length words in fixed-length sentences. Therefore, we propose an implicit control method, *Begin-Internal-End* (BIE) word-granularity embedding with its guided attention mechanism to adjust the word-level length format.

3.4.1 BIE Word-granularity Embedding

As shown in Figure 2 (left), each lyric token is added with a learnable embedding to record word length information[2], just like position embeddings, that is "Begin-Internal-End (BIE) word-granularity embedding". The design of BIE embedding symbols is inspired by the sequence tagging task (Huang et al., 2015). We use $[B - \{length\}]$, $[I - \{length\}]$, $[E - \{length\}]$ to indicate the beginning, inside, and end of a word or term (i.e., a combination of Chinese characters), and specifically splice the "BIE" mark with a number to record word length. For example, "$[B - 1]$" indicates the word length is 1, and "$[B-4]$, $[I-4]$, $[I-4]$, $[E-4]$" indicates the word length is 4. This labeling strategy can avoid word boundary confusion during training. We set additional embedding symbols $[S]$ (i.e., separator) and $[C]$ (i.e., count) to respectively correspond to the separator $[SEP]$ and sentence-level control characters, and $[O]$ (i.e., outside) to correspond to trigger words and the ending character $[END]$ in the lyric sequence.

Note that the lyric embedding sequence is not aligned with the BIE embedding sequence; it corresponds to the BIE embedding sequence shifted to the left. This setting aims to help the model learn to predict the word length of the next token for lyric sequence, and to learn when to stop sentence generation and feed new control characters.

3.4.2 Word-granularity Attention

The BIE word-granularity embeddings can only perceive the word length of the next lyric token in advance, but cannot predict the farther distance due to the unidirectional masked attention in autoregressive language model. When sentence length is fixed, BIE embeddings are difficult to reconcile and arrange the length of each word reasonably. Therefore, we design a word-granularity attention mechanism, which is guided by BIE embeddings, to perceive the word length information of all positions for the current token.

Concretely, the special decoder block with the word-granularity attention is placed at the bottom of the ChipSong model, on top of which the standard Transformer decoder block is stacked. The detailed structure is shown on the right of Figure 2 (right). The calculation process is as follows:

[2]Words length is obtained by the Chinese text segmentation tool, Jieba.

$$\hat{E}^{F_w} = E^{F_w} + E^P \quad (1)$$

$$C_w = Softmax(X'W_1\hat{E}^{F_w})\hat{E}^{F_w} \quad (2)$$

$$X_{out} = [X'; C_w]W_2 + X \quad (3)$$

First, the BIE embedding E^{F_w} and position embedding E^P are added to obtain $\hat{E}^{F_w}$, so that the BIE embedding sequence carries global position information. Then, after passing through the masked self-attention layer, a word-granularity attention layer is designed to compute the attention weights of the contextual lyric embeddings X' to the BIE embeddings $\hat{E}^{F_w}$, where a bilinear attention is applied, so as to obtain the contextual embedding C_w recording global words length for each lyric token. Finally, the contextual lyric embedding X' and global word-length embedding C_w are concatenated and pass through a linear layer to obtain fusion representation X_{out}, and a residual connection (He et al., 2016) is added to enhance the memory of the original input lyric embedding features X in decoder block. X_{out} is further modeled in subsequent Transformer decoder blocks.

3.5 Trigger Word Control

To produce enough trigger words during training to cover creators' input needs as much as possible, we adopt a two-stage strategy, establishing a candidate word list for each lyric in the first stage, and re-sampling the candidate list as trigger words during each training epoch in the second stage. Concretely, considering that general keyword extraction methods could result in a low coverage range of trigger words, all nouns, adjectives, and verbs[3] of the lyrics are reserved as the candidate word list after removing the stop words, and the candidate word list also preserves word frequency so that frequent words have a higher probability of being sampled. The number of trigger words sampled is determined according to the number of lyric sentences, and the rules are designed as follows:

$$\begin{cases} k = N_{sent}/2 - 1, & N_{sent} <= 12 \\ k = 5, & else \end{cases} \quad (4)$$

where k is the number of trigger words and N_{sent} is the number of sentences.

As shown in Figure 2, after building trigger words-lyric pair data, the trigger words sequence and lyric sequence are simply spliced and fed into the language model for training, guiding model self-supervisedly complements the lyric sequence according to partially-observed trigger word sequence, where trigger words are also separated by token "$[SEP]$". When prediction, feed trigger words, and the model complements the subsequent lyric part.

3.6 Rhyme Control

A paradigm of *reverse order training and shielding decoding* is designed to control rhymes. During training, we process the training data as inter-sentence normal order and intra-sentence reverse order, as shown in Figure 2. For example, when the original lyric sequence is "$..., [3], x_1, x_2, x_3, [SEP], [4], x_4, x_5, x_6, x_7, [SEP], ...$", it is transformed into "$..., [3], x_3, x_2, x_1, [SEP], [4], x_7, x_6, x_5, x_4, [SEP], ...$". In the same way, the BIE word-granularity embedding sequence is accordingly processed to keep consistency. The reverse order sentences input enables to learn a reverse language model, so that rhyming position is predicted first in a sentence, and the subsequent predictions coordinate the rhyming word for protecting the text fluency from being affected.

During prediction, the input pattern of "inter-sentence normal order, intra-sentence reverse order" is maintained; that is, the last word to rhyme in each sentence is predicted first, and then the rest of the sentence is predicted in reverse order. Then, a decoding shielding strategy is adopted to control the prediction of rhyming words. According to the Chinese 14-rhyme scheme, we build a rhyme dictionary whose key is the rhyme and value is the words that the rhyme matches. At the position of the last word in the sentence, the rhyme dictionary is queried according to the input rhyme, and the softmax output values corresponding to all non-rhymed words are reduced, so that the model selects outputs from rhyming words based on predicted probability distributions.

4 Experimental Setup

4.1 Data Preparation and Processing

We prepare a large lyric corpus to train the Chip-Song model. The lyric data is constructed with reference to ChineseLyrics[4], and we gather $848K$ Chinese popular lyrics, where the number of lyrics sentences is 27,181K, the average lyric length is

[3]POS-tagging information is obtained by the Jieba tool.

[4]https://github.com/dengxiuqi/ChineseLyrics

253 (excluding punctuation), the average number of sentences is 32, and the average length of each sentence is 8. To fit short-video durations, we tailor the lyrics into small segments of 8, 10, 12, 14, or 16 sentences in length, which considers that 8 to 12 lines of lyrics are generally required for a 60 sec. short-video song. Lyric tailoring also increases first line diversity. In addition, unsegmented lyrics are also incorporated as training data to preserve semantic integrity and learn long-range dependencies.

4.2 Evaluation Templates

To comprehensively evaluate the proposed system, we formulate multiple conditional templates for generation, which are provided in https://github.com/korokes/chipsong. Concretely, we build 15 groups of trigger words from users and construct 500 format templates from the format library, where the formats are extracted based on actual lyrics. Each format template is randomly assigned a rhyme pattern and a group of trigger words as a complete evaluation template for lyric generation. In addition, we sample the original lyrics corresponding to the format template to generate another group of trigger words as described in Section 3.5, which are only used to evaluate the effect of trigger words guiding content. The lyrics corresponding to these evaluation templates are eliminated from the training data.

For automatic evaluations, each template generates 20 samples, and a total of 500×20=10,000 samples are finally generated for evaluation. For human evaluations, 200 samples of 10 conditional templates are randomly reserved for evaluation.

4.3 Training settings

We use a 8-head, 8-layer, 512-dimensional Transformer to build the ChipSong model (39.5M). Actually, larger hidden layer dimensions can make the model perform better. For training, the Adam optimizer (Kingma and Ba, 2014) is used with an initial learning rate of 1.5e-4. The model is trained for about 3.5 days on two GTX 2080ti GPUs with a batch size of 8. The training data combines the segmented lyrics data and the raw lyrics data, eliminating lyrics corresponding to evaluation templates. Due to the large corpus and the duplication of segmented lyrics and original lyrics, we do not set too many training epochs, and set the training epochs to 3. Owing to sufficient lyrics gathering, we do not use the pretraining strategy. For prediction, we use Top$K$ decoding with a sampling value of 8.

4.4 Evaluation Metrics

Automatic Evaluation We use the trained lyric generation models on our corpus to evaluate the perplexity (PPL) and use the Distinct (MA-D1,D2, MI-D1,D2) metrics (Li et al., 2016) to evaluate the diversity of generated lyric texts. Moreover, we design the following metrics to evaluate the proposed conditional control ability: 1) sentence-level format accuracy (SA), the percentage of generated sentences with correct length. 2) word-level format accuracy (WA), the percentage of generated sentences whose words length arrangement is exactly the same as the label. 3) rhyme accuracy (RA), the percentage of generated sentences with correct rhymes. 4) word length accuracy (WA-N), the percentage of generated words containing N Chinese characters that are correct in position and length; as the Chinese word lengths are basically within 4, we evaluate the control accuracy of 1 to 5 word length. 5) trigger word effect, we first use trigger words extracted from the original lyrics to generate samples, and then use BLEU (Papineni et al., 2002) to compare content similarity between the original lyric and the generated lyric to evaluate the relevance of trigger words and contents indirectly.

Human Evaluation We recruit three postgraduates engaged in audio and music fields to score the generated lyrics on fluency, relevance, and listenability: (1) fluency (F), the quality of the generated lyrics, whether they are smooth, grammatical, and whether there are ill-formed sentences; (1=Bad to 3=Good). (2) relevance (R), the degree of relevance of the trigger words and the lyric content; (1=Bad to 3=Good). (3) listenability (L) (Watanabe et al., 2018), as lyrics, are the positions of words, lines, and segments natural? (1=Bad to 3=Good).

4.5 Baselines

For reference, a standard Transformer-based autoregressive language model (ALM) is trained as a baseline, with the same configuration as ChipSong. We also respectively train ALM-F, ALM-T, and ALM-R for observing a single conditional control effect, where the proposed control strategies of format (F), trigger word (T), and rhyme (R) are individually modeled into ALM for training (modeling all the three conditions into ALM is equal to ChipSong), with the same configuration

No.	Model	PPL (↓)	MA-D1	MI-D1	MA-D2	MI-D2	SA	WA	RA	WA-1	WA-2	WA-3	WA-4	WA-5
1	ALM	15.77	83.40	5.01	97.33	15.76	9.96	0.67	15.43	0.58	0.94	0.09	0.05	-
2	SongNet	12.33	88.05	4.77	98.05	18.92	97.80	5.89	13.82	3.67	6.82	0.75	0.59	-
3	ALM-F	8.49	89.24	4.47	**98.48**	17.39	98.38	**92.72**	16.44	**92.68**	**94.22**	**88.35**	**89.54**	**31.58**
4	ALM-T	14.78	86.27	3.49	97.20	14.84	10.68	0.51	12.01	0.45	0.59	0.07	-	-
5	ALM-R	15.55	**89.49**	4.76	98.28	19.63	9.57	0.36	98.38	0.35	0.46	0.08	-	-
6	ChipSong	**7.69**	89.20	**5.22**	98.04	**21.69**	**98.54**	86.64	**98.56**	86.04	89.03	78.74	76.11	18.42

Table 1: Automatic evaluation results of different models. SA: sentence-level format accuracy, WA: word-level format accuracy, RA: rhyme , WA-N: N-length word accuracy. Overall, ChipSong shows better control ability in all conditions. Note that single conditional control models perform better on corresponding conditions than ChipSong with full-conditional control applied, such as ALM-F on WA and WA-N metrics, because there are no constraints of other conditional controls, which is explained in result analysis.

No.	Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4
1	ALM	23.50	7.63	2.61	0.90
2	SongNet	25.41	6.25	1.88	0.61
3	ALM-F	26.29	7.88	2.69	1.03
4	ALM-T	**29.84**	**11.33**	**4.53**	**1.83**
5	ALM-R	23.83	6.11	1.75	0.54
6	ChipSong	28.55	9.56	3.38	1.35

Table 2: Effects of trigger words controlling contents.

as ChipSong. In addition, we compare ChipSong with SongNet (Li et al., 2020) that proposes a rigid format and rhyme control method. Due to differences in model settings and usage data, we use our data to retrain the SongNet model. For the models that lack the trigger word mechanism, the input is used as the first sentence and let the model continue to write like previous methods.

5 Experimental Results

5.1 Results

Table 1 and Table 2 show the experimental results of different models under each evaluation metrics. As can be seen from Table 1, ChipSong demonstrates good conditional control ability on format and rhyme, where the sentence-granularity format accuracy (SA) is 98.54%, the word-level format accuracy (WA) is 86.64%, and the rhyme accuracy (RA) is 98.56%. Since SongNet's rhyming modeling method cannot actively select rhymes and requires specific rhyming corpus for training, it isn't easy to exert its role in rhyming modeling to achieve good rhyming accuracy. ChipSong also demonstrates better PPL and generative diversity. Interestingly, the reverse order training of rhyme control (No.5) has little impact on the model PPL, indicating that the reverse language model still learns language rules. As shown in Table 2, ChipSong embodies better content control capabilities via the trigger word mechanism.

It can also be observed that a single conditional control model (No.3,4,5) generally performs better on its corresponding control condition because there are no constraints of the other two conditional controls. For example, in Table 1, without the constraints of trigger words and rhyme decoding shielding, ALM-F can focus more on controlling length format and obtain higher WA and WA-N scores, even achieving 92.72 on WA; in Table 2, without format and rhyme constraints, ALM-T has more opportunities to generate content related to trigger words for better BLEU scores.

5.2 Ablation

To further analyze the effect of each proposed conditional control, we respectively remove the conditional control of 1) word-level format control (WC); 2) sentence-level format control (SC); 3) trigger word control (TC); 4) rhyme control (RC) to train the ablation models. Two internal structures of WC, 6) BIE embedding in WC (WC-Emb) and 7) word-granularity attention in WC (WC-Att), are also ablated for evaluation.

The experimental results are shown in Table 3 and Table 4. As can be observed from the tables, format (No.2,3,4,5) and rhyme control (No.7) increase the diversity of generation while trigger word control (No.6) decreases the diversity. The modeling of word-level format control, WC, WC-Att, and WC-Emb, plays an important role in reducing the PPL. When the modeling of WC, SC, RC, or TC is removed separately, the accuracy of the corresponding evaluations, SA, WA, RA, or BLEU, is obviously reduced, indicating the effectiveness of the proposed control methods. Although WC-Att and WC-Emb both play a positive role in word-level format control, trigger word control (TC) and rhyme control (RC) have a negative effect on word-

No.	Ablation Model	PPL (↓)	MA-D1	MI-D1	MA-D2	MI-D2	SA	WA	RA	WA-1	WA-2	WA-3	WA-4	WA-5
1	ChipSong	7.69	89.20	5.22	98.04	21.69	98.45	86.64	98.56	86.04	89.03	78.74	76.11	18.42
2	w/o WC-Emb	8.00	88.40	5.18	97.21	22.62	98.04	79.48	98.54	79.67	83.43	62.38	56.22	7.89
3	w/o WC-Att	9.75	88.89	4.94	97.90	21.53	97.98	78.95	98.53	77.94	82.75	66.88	58.01	15.79
4	w/o WC	12.55	85.51	4.11	96.65	15.99	98.36	5.39	98.22	3.79	6.55	0.78	0.36	-
5	w/o WC, SC	14.50	87.00	4.69	98.05	14.84	9.40	0.53	97.68	0.72	1.05	0.13	0.07	-
6	w/o TC	8.37	91.16	5.73	98.95	23.07	98.31	89.86	98.87	90.09	91.94	83.07	84.68	21.05
7	w/o RC	7.84	87.68	4.85	97.34	18.30	98.75	89.37	15.19	88.36	91.10	83.48	79.86	28.95

Table 3: Ablation results. WC: word-level format control, SC: sentence-level format control, TC: trigger word control, RC: rhyme control. Ablating one control of ChipSong causes the corresponding evaluation score to decrease, while evaluation scores of other controls increase due to the reduction of constraints for generation.

No.	Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4
1	ChipSong	28.55	9.56	3.38	1.35
2	w/o WC-Emb	27.97	8.98	3.10	1.18
3	w/o WC-Att	26.93	8.01	2.67	1.02
4	w/o WC	27.49	8.39	2.81	1.00
5	w/o WC, SC	25.38	10.28	4.57	2.02
6	w/o TC	24.85	6.26	1.83	0.63
7	w/o RC	31.23	12.69	5.58	2.49

Table 4: Ablation results of trigger words controlling contents.

No.	Model	F	R	L	Avg
1	ALM	2.51	2.07	2.61	2.40
2	SongNet	2.53	1.82	2.96	2.44
3	ChipSong	2.56	2.44	2.96	2.65
4	ChipSong w/o WC	2.48	2.47	2.95	2.63
5	ChipSong w/o WC, SC	2.46	2.67	2.75	2.63
6	ChipSong w/o TC	2.51	1.75	2.97	2.41
7	ChipSong w/o RC	2.54	2.60	2.95	2.70

Table 5: Results of human evaluations. F: Fluency; R: Relevance; L: Listenability. Avg is the average score of F, R and L.

level format control, where the scores of WA and WA-N rise when TC or RC is ablated.

5.3 Human Evaluation

Table 5 shows the experimental results of human evaluation in fluency (F), relevance (R), and Listenability (L). On the whole, the fluency scores of the models are not much different (2.46-2.56 points), which is attributed to sufficient corpus for training. ChipSong scores far higher than baselines (No.1,2) in relevance evaluation due to the modeling of trigger word mechanism. It can also be seen that when the control of rhyme or format (No.4,5,7) is lifted, the relevance is improved; we conjecture that the model has more opportunities to generate related content when the format or rhyme is not restricted. ChipSong w/o RC (No.7) gain the best average score; this is because our human evaluation does not consider the evaluation of lyric rhymes. The listenability scores of the models without format control (No.1,5) drop from nearly full marks, because the free generation is prone to generate too short or too long sentences, or two consecutive sentences with large length differences, which is not conducive to fit songs.

5.4 Trigger word coverage

We count the sampled trigger words and the sampled trigger words without repetition in training, which aims to observe the trigger word coverage. The results are shown in Figure 3. Due to the two-stage strategies, a large number of trigger words are produced for training. The number of sampling words is 3.98e7, and is only 3.16e5 after deduplication. As shown in Figure 3, as the extracted trigger words increase, new trigger words increase very little, which shows that our method covers a relatively comprehensive range of trigger words to handle out-of-distribution and cover the general input needs for users.

5.5 Case Analysis

As shown in Figure 4, we enumerate some generated lyrics of the ChipSong system in several scenarios: 1) a Chinese Hanmai song with a specific format; 2) customizing format according to a song; 3) imitative writing a lyric for a cover song version, where the sentence- and word-level format are extracted from the original lyric, Han Hong's "Qingchun". For the first case in each template in the figure, we also provide the English translation for understanding the generated lyrics. For the first template in a given word-granularity length format, we provide a human-annotated word segmentation boundary with the green vertical line |. As can be seen from the generated results, ChipSong can fine-grainly adjust the generated lyrics' format to adapt to any song, render the content guided

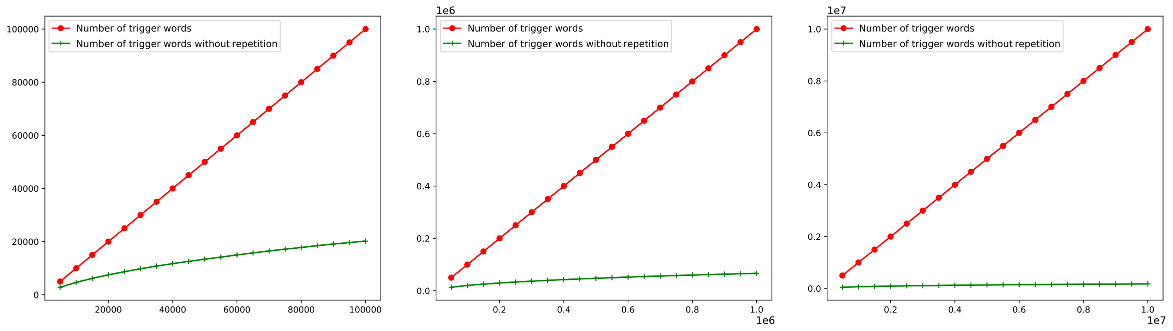

Figure 3: The count of sampled trigger words. Three figures use different scales. The total sampling number is 3.98e7, and is only 3.16e5 after deduplication. New trigger words increase very little as the increase of sampling trigger words, indicating that sampled trigger words cover a relatively comprehensive range to handle input needs.

Format	Trigger word (T) & Rhyme (R) & Generated lyric			English Translation of the first case

Sentence granularity:
7;7;7;7;7;7;7;7
Word granularity:
2,2,1,2;
2,1,1,1,2;
2,2,1,2;
1,1,2,1,2;
2,1,1,1,2;
1,1,1,2,1,1;
2,2,2,1;
1,1,1,1,1,2
	T: 今朝有酒今朝醉
	R: ui, ei
	良宵|不醉|亦|不归 (ui)
	往日|与|你|共|举杯 (ei)
	未知|哪日|再|相会 (ui)
	只|为|来生|永|跟随 (ui)
	再多|苦|我|也|无悔 (ui)
	哥|只|想|努力|去|追 (ui)
	人生|难得|几回|醉 (ui)
	要|喝|倒|咱|再|举杯 (ei)
		T: 今朝有酒今朝醉
		R: ui, ei
		不管|荣华|与|富贵 (ui)
		只想|与|你|干|一杯 (ei)
		走遍|千山|和|万水 (ui)
		让|我|品尝|这|滋味 (ei)
		女孩|你|让|我|心醉 (ui)
		且|看|那|远方|多|美 (ei)
		波涛|万里|长江|水 (ui)
		风|又|吹走|了|疲惫 (ei)
			T: 今朝有酒今朝醉
			R: ui, ei
			痴心|爱过|这|一回 (ui)
			如果|爱|你|学不会 (ui)
			当|情|已经|被|摧毁 (ui)
			还能|怎么|去|挽回 (ui)
			梦里|是|你|的|余味 (ei)
			望着|那|身影|才|美 (ei)
			长夜|不知|何时|归 (ui)
			我|又|为|谁|在|沉醉 (ui)
				T: If there is liquor now, then drink now
				R: ui, ei
				Drink until thoroughly drunk in a good night
				I raised the glass with you in bygone days
				Do not know when will see you again
				Hope is always with you in future life
				No matter how difficult it is, no regrets
				Try to pursue is the only thing I want
				Life is rarely a few times drunk
				Drink a toast again, just to get drunk

Sentence granularity:
10;8;10;8;10;8;10;8
Word granularity:
User-defined or complementing from the library
	T: 玫瑰
	R: ui, ei
	那天你轻轻的拿着玫瑰(ui)
	破碎的心日渐枯萎(ei)
	告诉我温柔到底给了谁(ui)
	最冷时有谁来安慰(ei)
	不愿为我最后流一滴泪(ei)
	有多少人今生无悔(ui)
	就要说爱情能够再轮回(ui)
	都说痴情叫人憔悴(ui)
		T: 夏天
		R: an, uan, ian
		当樱花雨落在我的夏天(ian)
		说好一起直到永远(uan)
		夏日微风轻吻你们的脸(ian)
		你总是笑得那么甜(ian)
		绚烂白玫瑰开在心里边(ian)
		让誓言永远不改变(ian)
		爱像阵清风吹过我胸前(ian)
		飘进了片片彩云间(ian)
			T: 红尘,流水,人世
			R: ang, iang, uang
			笑看落花流水一如往常(ang)
			道不尽我此生轻狂(uang)
			只身凡间游荡江湖渺茫(ang)
			风花雪月你在何方(ang)
			是谁倩影拂过一夜微凉(iang)
			是否能将红尘遗忘(ang)
			经年里看遍这人世匆忙(ang)
			任谁淡墨染了云裳(ang)
				T: Rose
				R: ui, ei
				You gently held the rose that day
				The wounded heart slowly withers
				Tell me who you gave your tenderness to
				When it's cold, who is there to comfort me
				Do not want to shed a single tear for me
				How many people have no regrets in this life
				Love can make life reincarnate
				They say infatuation makes people gaunt

(Extracting format from original lyric)
也许不会再看见
离别时微黄色的天
有些人注定不会再见
那些曾青涩的脸
我拿起棕榈树的叶子
放在青涩的石板前
祭奠那些流逝的青春
和曾懵懂的誓言
	T: 恋爱
	R: an, uan, ian
	星河日月共缠绵(ian)
	清风拂去多少流年(ian)
	不必问世间情缘深浅(ian)
	江湖路尽头又见(ian)
	你看那山高水也长远(uan)
	时光穿行在云水间(ian)
	刻下一张不老的容颜(an)
	随你海角到天边(ian)
		T: 酒
		R: ui, ei
		今朝拂尘再一醉(ui)
		从此天涯远去不归(ui)
		相爱过只求一生无悔(ui)
		身旁的恋人是谁(ui)
		让我在除夕夜又梦回(ui)
		些许诗意比落花美(ei)
		悠悠流水情匆似流水(ui)
		等你滴入我心扉(ei)
			T: 醉红颜, 酒
			R: an, uan, ian
			彼时把酒忆当年(ian)
			记得那年桃花初见(ian)
			或许你再续前生情缘(uan)
			如今我轻语谁言(an)
			人却在转瞬间霜满天(ian)
			听到耳边箫声渐远(uan)
			独饮一杯浊酒醉红颜(an)
			都说往事化青烟(an)
				T: In love
				R: an, uan, ian
				Stars, river, sun, and moon lingering together
				The breeze blows, took away the years
				Do not need to ask depth of love in the world
				At the end of the road, we will meet
				Look at the high mountains and far water
				Time travels between clouds and water
				Carve an ageless appearance of beauty
				To the ends of the earth with you

Figure 4: Cases of lyrics generated by ChipSong. The generated results are marked in blue font. The rhymes are marked in grey font. T: Trigger words; R: Rhyme. For the fist template, the green vertical line | is used to manually annotate word segmentation boundaries. For the first case in each template, the English translation is given.

by open-ended trigger words, and maintain the rhyme. More generated cases are provided in https://github.com/korokes/chipsong.

6 Conclusion

In this work, we develop *ChipSong*, a lyric generation system, to assist musical short-video creators for **Chi**nese **p**opular **song** creation. ChipSong fine-grainly adjusts lyric generation to meet the creator's needs in various scenarios via the proposed strategies of sentence- and word-level format control, trigger word control, and rhyme control. In the future, we would like to consider melody generation and other attributions control of lyrics.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Haoshen Fan, Jie Wang, Bojin Zhuang, Shaojun Wang, and Jing Xiao. 2019. A hierarchical attention based seq2seq model for Chinese lyrics generation. In *Proceedings of the Pacific Rim International Conference on Artificial Intelligence*, pages 279–288.

Zhipeng Guo, Xiaoyuan Yi, Maosong Sun, Wenhao Li, Cheng Yang, Jiannan Liang, Huimin Chen, Yuhui Zhang, and Ruoyu Li. 2019. Jiuge: A human-machine collaborative Chinese classical poetry generation system. In *Proceedings of the 57th Annual Meeting of the Association for Computational*

Linguistics: System Demonstrations, pages 25–30, Florence, Italy. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Jinyi Hu and Maosong Sun. 2020. Generating major types of Chinese classical poetry in a uniformed framework. *CoRR*, abs/2003.11528.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

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

Hsin-Pei Lee, Jhih-Sheng Fang, and Wei-Yun Ma. 2019. iComposer: An automatic songwriting system for Chinese popular music. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 84–88.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Piji Li, Haisong Zhang, Xiaojiang Liu, and Shuming Shi. 2020. Rigid formats controlled text generation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 742–751.

Xu Lu, Jie Wang, Bojin Zhuang, Shaojun Wang, and Jing Xiao. 2019. A syllable-structured, contextually-based conditionally generation of Chinese lyrics. In *Proceedings of the Pacific Rim International Conference on Artificial Intelligence*, pages 257–265.

Enrique Manjavacas, Mike Kestemont, and Folgert Karsdorp. 2019. Generation of hip-hop lyrics with hierarchical modeling and conditional templates. In *Proceedings of the International Conference on Natural Language Generation*, pages 301–310.

Nikola I Nikolov, Eric Malmi, Curtis Northcutt, and Loreto Parisi. 2020. Rapformer: Conditional rap lyrics generation with denoising autoencoders. In *Proceedings of the International Conference on Natural Language Generation*, pages 360–373.

Hugo Gonçalo Oliveira, Raquel Hervás, Alberto Díaz, and Pablo Gervás. 2017. Multilingual extension and evaluation of a poetry generator. *Natural Language Engineering*, 23(6):929–967.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the annual meeting on association for computational linguistics (ACL)*, pages 311–318. Association for Computational Linguistics.

Peter Potash, Alexey Romanov, and Anna Rumshisky. 2015. Ghostwriter: Using an LSTM for automatic rap lyric generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1919–1924.

Peter Potash, Alexey Romanov, and Anna Rumshisky. 2018. Evaluating creative language generation: The case of rap lyric ghostwriting. In *Proceedings of the Second Workshop on Stylistic Variation*, pages 29–38.

Liang-Hsin Shen, Pei-Lun Tai, Chao-Chung Wu, and Shou-De Lin. 2019. Controlling sequence-to-sequence models-a demonstration on neural-based acrostic generator. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, pages 43–48.

Zhonghao Sheng, Kaitao Song, Xu Tan, Yi Ren, Wei Ye, Shikun Zhang, and Tao Qin. 2021. SongMASS: Automatic song writing with pre-training and alignment constraint. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13798–13805.

Kento Watanabe, Yuichiroh Matsubayashi, Satoru Fukayama, Masataka Goto, Kentaro Inui, and Tomoyasu Nakano. 2018. A melody-conditioned lyrics language model. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 163–172.

Lanqing Xue, Kaitao Song, Duocai Wu, Xu Tan, Nevin L Zhang, Tao Qin, Wei-Qiang Zhang, and Tie-Yan Liu. 2021. DeepRapper: Neural rap generation with rhyme and rhythm modeling. *arXiv preprint arXiv:2107.01875*.

Rui Yan, Cheng-Te Li, Xiaohua Hu, and Ming Zhang. 2016. Chinese couplet generation with neural network structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2347–2357, Berlin, Germany. Association for Computational Linguistics.

Rongsheng Zhang, Xiaoxi Mao, Le Li, Lin Jiang, Lin Chen, Zhiwei Hu, Yadong Xi, Changjie Fan, and Minlie Huang. 2020a. Youling: An ai-assisted lyrics creation system. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 85–91.

Yuchao Zhang, Pengmiao Li, Zhili Zhang, Bo Bai, Gong Zhang, Wendong Wang, Bo Lian, and Ke Xu. 2020b. Autosight: Distributed edge caching in short video network. *IEEE Network*, 34(3):194–199.

Hongyuan Zhu, Qi Liu, Nicholas Jing Yuan, Chuan Qin, Jiawei Li, Kun Zhang, Guang Zhou, Furu Wei, Yuanchun Xu, and Enhong Chen. 2018. Xiaoice band: A melody and arrangement generation framework for pop music. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2837–2846.