RL-Chord: CLSTM-Based Melody Harmonization Using Deep Reinforcement Learning

Shulei Ji, Xinyu Yang, Jing Luo, and Juan Li

***Abstract*— Automatic music generation is the combination of artificial intelligence and art, in which melody harmonization is a significant and challenging task. However, previous recurrent neural network (RNN)-based work fails to maintain long-term dependency and neglects the guidance of music theory. In this article, we first devise a universal chord representation with a fixed small dimension, which can cover most existing chords and is easy to expand. Then a novel melody harmonization system based on reinforcement learning (RL), RL-Chord, is proposed to generate high-quality chord progressions. Specifically, a melody conditional LSTM (CLSTM) model is put forward that learns the transition and duration of chords well, based on which RL algorithms with three well-designed reward modules are combined to construct RL-Chord. We compare three widely used RL algorithms (i.e., policy gradient, *Q*-learning, and actor–critic algorithms) on the melody harmonization task for the first time and prove the superiority of deep *Q*-network (DQN). Furthermore, a style classifier is devised to fine-tune the pretrained DQN-Chord for zero-shot Chinese folk (CF) melody harmonization. Experimental results demonstrate that the proposed model can generate harmonious and fluent chord progressions for diverse melodies. Quantitatively, DQN-Chord achieves better performance than the compared methods on multiple evaluation metrics, such as chord histogram similarity (CHS), chord tonal distance (CTD), and melody–chord tonal distance (MCTD).**

***Index Terms*— Deep reinforcement learning (RL), long short-term memory, melody harmonization with chords, symbolic music generation.**

I. Introduction

GENERATING music works that are comparable to human creations using artificial intelligence techniques is an effortful task. Automatic music generation contains assorted subtasks [1], [2], such as melody generation [3], [4], polyphony generation [5], [6], and multitrack generation [7], [8]. Among them, automatic accompaniment generation [9] is a conditional generation task, which can enrich and support monotonous melody expression and make music more plump. Generally, two kinds of accompaniments can be concluded, i.e., multitrack accompaniment (or arrangement) [10] and chord accompaniment [9], [11]. This article focuses on the latter, which is also named as melody harmonization.

Previous neural-network-based melody harmonization studies mostly used recurrent neural networks (RNNs). Lim et al. [11] first used bidirectional long short-term memory (BLSTM) to predict a chord label for each bar from 24 major/minor triads. Yeh et al. [9] extended this method to predict not only the chord label but also the chord function. The generated chord types are twice as many as before, but the number of chords per bar is still limited to two at most. These methods focus more on learning the correspondence between the melody and the chord accompaniment, either ignoring the chord progression rules or leaning the chord function that is not suitable for all music. Despite the effectiveness of previous studies, the type and number of chords generated by these methods are still restricted, and the generation process does not integrate external music theory knowledge. Moreover, previous studies all train the chord generation models in a supervised manner, which cannot be adapted to scenarios lacking melody–chord parallel datasets.

Most of the existing melody harmonization models use maximum likelihood estimation (MLE), which brings following substantive problems. First, the MLE-based model tends to remember common patterns in the corpus, leading to the generated music losing some diversity and creativity. Standard MLE training also gives rise to the "exposure bias" problem due to the difference between training and testing stages [12]. In addition, the MLE-based models will raise two loss evaluation mismatch problems [13]. One is evaluation granularity mismatch. When evaluating, music experts tend to analyze the whole phrase, while MLE optimizes only note-level or chord-level loss, failing to have a broader vision of generated music. The other is criteria mismatch. The optimization strategies such as cross-entropy (CE) are usually used during model training, but the generated results are evaluated on non-differentiable and discrete metrics during testing. Introducing music evaluation metrics during model training will fundamentally solve the criteria mismatch problem.

Recent studies [14], [15] have shown that the above issues can be alleviated to some extent by combining reinforcement learning (RL) techniques. Remarkable progress has been made in the field of text generation [16], image generation [17], etc. As for music generation, some researchers [18], [19], [20], [21] have begun to adopt RL algorithms as well and achieved pretty good results. Reinforcement learning with long-term

Manuscript received 23 July 2021; revised 11 October 2022; accepted 20 February 2023. *(Corresponding author: Juan Li.)*

Shulei Ji, Xinyu Yang, and Jing Luo are with the School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049, China (e-mail: taylorji@stu.xjtu.edu.cn; yxyphd@mail.xjtu.edu.cn; luojingl@stu.xjtu.edu.cn).

Juan Li is with the Music Education Center, Xi'an Jiaotong University, Xi'an 710049, China (e-mail: lijuan@xjtu.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2023.3248793.

Digital Object Identifier 10.1109/TNNLS.2023.3248793

perspectives is very suitable for music generation. Especially for melody harmonization, not only the concordance between the melody notes and chords but also the chord progression and global structure need to be considered. Typically, the RL-based methods regard music generation as a sequential decision problem. The music element (note, chord, etc.) to be predicted in the next step is the next action to be taken, and the instant reward function is customized according to the specific task. Although RL has shown effectiveness in the field of automatic music generation, its application is still relatively rare compared with other methods. In terms of chord generation, only a few researchers adopted simple RL algorithms [22] resulting in somewhat monotonous generation results.

Improving the diversity, harmonicity, and fluency of generated chord accompaniments while mitigating problems caused by standard MLE training is our goal. In this article, we propose an RL framework for melody harmonization with chords (RL-Chord). A novel universal chord representation with a 20-D multi-hot (MH) vector is devised to represent various chords. The melody notes and chords are represented as one-to-one correspondence to avoid mandatory chord durations. The underlying model of RL-Chord is a two-layer conditional LSTM (CLSTM), where a melody condition vector and the previous chords are taken as input to predict the next chord autoregressively. Three reward modules are designed for RL-Chord to describe high-quality chords, which incorporate some external knowledge such as music interval consonance, mutual information theory, and chord progression rules. We also construct a style classifier to fine-tune the pretrained DQN-Chord, thereby generating chords in Chinese folk (CF) style unsupervisedly. We will show that RL-Chord composed of the above techniques can produce better generation results.

Our main contributions can be summarized as follows.

1) We propose a novel melody harmonization approach with an improved CLSTM model and a newly designed chord representation.
2) We construct RL-Chord by combining RL with CLSTM and compare three typical RL algorithms [i.e., policy gradient (PG), Q-learning, and actor–critic (AC)] on the melody harmonization task for the first time.
3) Experimental results on two datasets show that DQN-Chord has a pretty improvement over the baseline in terms of objective metrics, human evaluation, and qualitative analysis.

The rest of this article is organized as follows. In Section II, we discuss the related work about music representation, melody harmonization, and RL-based music generation. Section III elaborates the proposed chord generation method, including chord representation and CLSTM. In Section IV, we devise comprehensive reward modules and put forward RL-Chord by applying three RL algorithms on CLSTM. Section V demonstrates the experimental results. And the last section concludes this article and presents some possible future work.

II. Related Work

A. Music Representation

Most of the previous studies [9], [11], [23] encoded chords as one-hot (OH) vectors and paid more attention to the triads. The OH representation brings a problem that more chord types will lead to sparser representations thus increasing the difficulty of model learning. Other studies [24], [25] directly used 12-D MH vector to depict the chord pitches in the chromatic scale, which ignores the octave where the chord is located and cannot determine the order of chord pitches in the inference stage. The representation proposed in this article can represent any number of chord types without increasing the representation dimension, indicate the octave where a chord is located explicitly, and generate different inversions of one chord.

Some studies [26], [27], [28] have represented a note as a tuple containing several attributes (such as pitch, duration, and velocity). Hsiao et al. [29] proposed the compound word representation, which groups the neighboring tokens that define a musical event into a super token and predicts multiple tokens at once through different feedforward heads. Inspired by this, this article combines four properties of the chord into a super token and encodes it into an MH vector. Four properties are predicted simultaneously by four fully connected layers.

B. Melody Harmonization

Several machine learning methods are proven to be effective in melody harmonization, such as genetic algorithm (GA) [9], [30], probability graph model (PGM) like hidden Markov model (HMM) [31], [32], and probabilistic context-free grammar (PCFG) [33]. The GA only applies the melody and chord information within a short segment to calculate the fitness function. The PGM predicts chords solely depending on the current state, ignoring the surrounding music information. As a result, these methods generally cannot maintain long-term dependence in the generated chord progression.

With the development of deep neural networks, multiple RNN-based melody harmonization models have been born. Lim et al. [11] used a two-layer BLSTM to learn the correspondence between melody and chord sequence pairs, which achieves better results than the HMM models. Yeh et al. [9] extended the BLSTM model to predict not only chord labels but also chord functions, and they concluded that the results of deep-learning-based methods are closer to the human-composed than the non-deep learning approaches. Sun et al. [23] took the melody and partially masked chord sequences as the input of a BLSTM-based network to learn the missing parts of chord sequences. Referring to the idea of Coconet [34], they used blocked Gibbs sampling in the inference stage.

The above work limits the types of generated chords and the number of chords per bar. The BLSTM-based methods concentrate more on learning the correspondence between melody and chord ignoring the previously generated chords and the learning of chord transitions. Although RNNs can generate chords with some long-term consistency, the whole training process lacks the guidance of music theory knowledge, which solely learns the data distribution to generate statistically significant content. The method proposed in this article will alleviate the above problems.

C. RL for Music Generation

Reinforcement learning has shown its power in music generation, which need not learn the given labels and pays more attention to the long-term returns under action sequences. In 2017, drawing on the reinforcement theory, Yu et al. [18] first proposed a discrete sequence generation framework named SeqGAN and proved its performance in the music generation task. Jacques et al. [19] devised a novel sequence learning method, RL Turner, by combining maximum likelihood (ML) and RL, whose rewards are derived from the pretrained RNN and music theory rules. Zhang et al. [26] designed a discriminator based on the self-attention mechanism to calculate the reward for each generated step (REGS) effectively. Jin et al. [35] put forward a style-specific music composition neural network (MCNN) by combining GAN and AC network, where the AC network is to fine-tune LSTM music generator, and the reward function is composed of the probability output of CNN discriminator and music rules.

In terms of conditional music generation, Jiang et al. [20] proposed the RL-Duet for online accompaniment generation, realizing real-time interaction with human input with no delay. The accompaniment here refers to another melody line. The core of RL-Duet is a valid reward model considering the compatibility of the generated notes with both the interpart and intrapart contexts. Afterward, Jiang et al. [21] proposed an online counterpart generation system FolkDuet for given CF melodies, which contains two reward functions for modeling the counterpart interaction and learning the CF style, respectively. Unlike the above studies, our research task focuses more on chord accompaniments. To our knowledge, only Shipra and Banka [22] used the Q-learning to generate simple major triad chords based on the basic music theory. This article compares chord generation models based on three RL frameworks for the first time.

III. Proposed Method

A. Music Representation

We encode each chord into an MH vector, which consists of four parts: rest, root note position, chord inversion type, and chord pitch set. The definition and encoding way of each part are introduced as follows.

1) *Rest:* Indicates whether there is a chord playing. It is represented by a 1-D binary vector, where 1 means rest, and 0 means playing;
2) *Root Note Position (rnp):* Represents the octave where the chord root note is located. The root notes involved in this article only exist within two octaves, thereby the rnp is represented by a 2-D OH vector;
3) *Chord Inversion Type (cit):* Depicts the inversion type of the chord. There are four chord inversion types in this article, i.e., root position, first inversion, second inversion, and third inversion. Therefore, this part is represented by a 4-D OH vector. Fig. 1(a) shows how we encode the chord inversion type with a seventh chord as an example;
4) *Chord Pitch Set (cps):* Represents the pitch set that makes up the chord. The pitches are normalized into the chromatic scale. The cps is encoded into a 13-D MH vector, and the last dimension is set to 1 and 0 when representing the triads and sevenths, respectively.

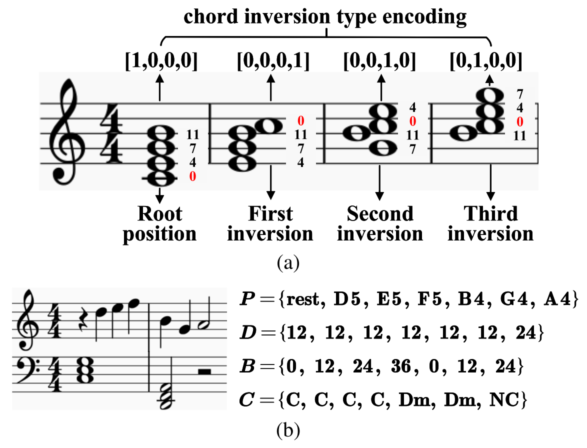

chord inversion type encoding
[1,0,0,0]
[0,0,0,1]
[0,0,1,0]
[0,1,0,0]
11 7 4 0
0 11 7 4
4 0 11 7
7 4 0 11
Root position
First inversion
Second inversion
Third inversion
(a)
P = {rest, D5, E5, F5, B4, G4, A4}
D = {12, 12, 12, 12, 12, 12, 24}
B = {0, 12, 24, 36, 0, 12, 24}
C = {C, C, C, C, Dm, Dm, NC}
(b)

Fig. 1. Music representation. (a) Chord inversion-type representation. The chord pitch set in the chromatic scale is $\{0, 4, 7, 11\}$. We take the position of the minimum pitch as the mark of the chord inversion type. In this case, [1,0,0,0] encoding of chord inversion type means the root position; [0,0,0,1] represents the first inversion; [0,0,1,0] represents the second inversion; [0,1,0,0] represents the third inversion. (b) Sequence-based music representation, where P is the pitch sequence, D is the duration sequence, B is the within-bar position sequence, and C is the chord sequence. The four sequences have the same length.

Finally, each chord is represented by a 20-D binary vector. It is noteworthy that only the first dimension is set to 1 when no chord is playing (NC). Our proposed representation hardly expands the dimension of chord representations even when the number of chord types increases. Besides, taking into account the octave of the root note and the chord inversion type makes the generation goal clearer.

Melody is represented in the same way as FolkDuet [21], namely, sequence-based representation. Each note is depicted by its pitch P, duration D, and within-bar position B, which are encoded by OH vectors, respectively. In this article, the pitch range is from 48 to 95 (C3-B6), a total of 49 (including rest). The quarter note is quantified into 12, and the duration range is from 2 to 48. The within-bar position helps the agent learn the chord onsets per bar, and a bar with a 4/4 time signature theoretically has 48 optional chord onsets. Compared with the time quantization method (such as piano roll), this representation is more compact and avoids the token imbalance problem.

We observe from the datasets that the onsets of chords usually coincide with that of melody notes, and the beginning of bars is frequently accompanied by that of chords. The alignment statistics of the onsets of chords, melody notes, and bars are shown in Table I. The alignment ratio of the chords to notes is calculated as the number of chords that are aligned with the onsets of melody notes divided by the total number of chords. The number of bars starting with a chord divided

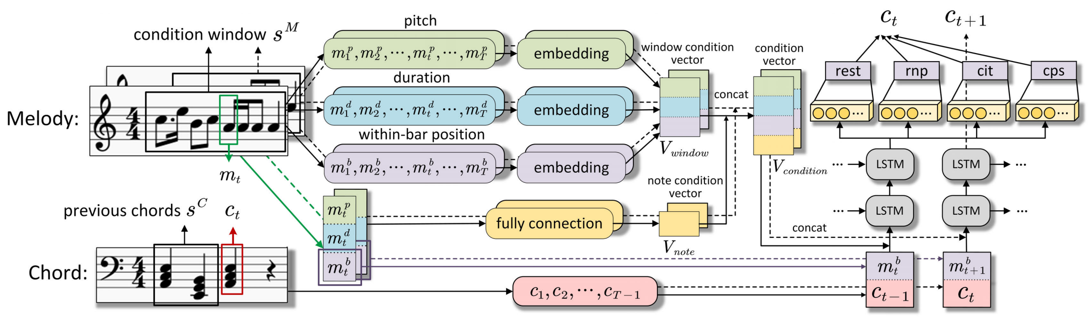


Fig. 2. Framework of the proposed CLSTM. The melody segment in the condition window s^M is used to construct the condition vector $V_{\text{condition}}$ as the additional input at each step, which consists of the window condition vector V_{window} and the note condition vector V_{note}. The pitch sequence, duration sequence, and within-bar position sequence are fed into three embedding layers to form V_{window}, and the note information at the current time is concatenated to form V_{note}. At each step, the output of CLSTM is fed into four fully connected layers to represent four parts making up a chord.

TABLE I
ALIGNMENT STATISTICS ON TWO DATASETS

Data set	Alignment ratio	
	chords to notes	bars to chords
NMD	1.0000	0.9374
Wiki	0.9523	0.9944

by the total number of bars is the alignment ratio of bars to chords. The statistical results in the table are consistent with our observations. Thus, we attempt to represent the chords and melody notes as one-to-one correspondence regardless of the chord duration. The rare non-aligned cases are solved through simple data processing like splitting notes or adding rest. The sequence-based representation of melody and chord is shown in Fig. 1(b).

B. Problem Definition

We will explain the notations appearing in this article and formulate the task to be solved briefly. In short, the task is to harmonize a given melody with appropriate chords. We denote the melody sequence as $M = \{m_1, m_2, \ldots, m_T\}$, the chord sequence as $C = \{c_1, c_2, \ldots, c_T\}$, and T represents the length of the sequence. Since the previous assumption of the one-to-one relationship between melody and chord, the melody has the same sequence length as the chord. The melody sequence can be divided into pitch, duration, and within-bar position sequences, which are expressed as $M^p = \{m_1^p, m_2^p, \ldots, m_T^p\}$, $M^d = \{m_1^d, m_2^d, \ldots, m_T^d\}$, and $M^b = \{m_1^b, m_2^b, \ldots, m_T^b\}$, respectively. When generating the chord at time t, we adopt a sliding window of length L over the melody sequence to select more valuable condition. The sliding window covers the melody notes before and after time t.

We regard melody harmonization with chords as an RL problem in the follow-up. The current state s_t is composed of previous chords $s_t^C = c_{1:t-1}$ and the current melody conditions $s_t^M = m_{t-L/2:t+L/2-1}$. Generating the corresponding chord c_t for the current melody note m_t is treated as taking an action a_t, then an instant reward r_t is calculated for the action, and the state s_{t+1} is updated by the generated chord. This continues iteratively until the whole chord progression is completed.

C. CLSTM

We improve the previous LSTM-based chord generation models [9], [11] and propose the conditional LSTM (CLSTM), as shown in Fig. 2. To learn the chord transition rules, we feed the previous chord c_{t-1} into model at each step and concatenate it with the within-bar position of the current note m_t^b to learn the onset and duration of the chord better. The main input of CLSTM is calculated as follows:

$$\text{input}_t^{c+b} = \text{concat}\left(c_{t-1}, m_t^b\right) \tag{1}$$

where concat represents the concatenation of vectors.

The conditional mechanism is embodied in that we compute a condition vector $V_t^{\text{condition}}$ using melody information at each step to guide the chord generation. So the input vector is computed as

$$\text{input}_t = \text{fc}\left(\text{concat}\left(\text{input}_t^{c+b}, V_t^{\text{conditon}}\right)\right) \tag{2}$$

where fc represents the fully connected layer.

1) Condition Vector: Whether a chord is concordant is typically judged by its nearby melody notes and the influence of the melody notes far away from it is negligible. Accordingly, we use a fixed-length sliding window to select the melody conditions. Specifically, we regard the current chord position as the midpoint and clip the melody condition $m_{t-L/2:t+L/2-1}$ contained in the window of length L. The condition vector comprises two parts, the window condition vector V_{window} and the note condition vector V_{note}, respectively. The former makes use of all the note information within the window. The pitch, duration, and within-bar position sequences are fed into three embedding layers, whose outputs are concatenated and fed into a fully connected layer to produce V_{window}

$$\begin{aligned} k_{\text{embed}} &= \text{embedding}_k\left(m_{t-L/2:t+L/2-1}^k\right), \quad k \in \{p, d, b\} \\ V_{\text{window}} &= \text{fc}(\text{concat}(p_{\text{embed}}, d_{\text{embed}}, b_{\text{embed}})) \end{aligned} \tag{3}$$

where embedding represents the embedding layer. The note condition vector V_{note} makes use of only the note information at the current step. The pitch, duration, and within-bar position of the present melody note are concatenated directly, and the concatenated result is input into a fully connected layer to obtain V_{note}

$$V_{\text{note}} = \text{fc}\left(\text{concat}\left(m_t^p, m_t^d, m_t^b\right)\right). \tag{4}$$

The final condition vector is the result of concatenating V_{window} and V_{note}

$$V_{\text{condition}} = \text{concat}(V_{\text{window}}, V_{\text{note}}). \tag{5}$$

2) Joint Optimization: Due to the previously proposed nontrivial chord representation, we feed the output of LSTM into four fully connected layers to generate four vectors output_k ($k \in \{\text{rest}, \text{rnp}, \text{cit}, \text{cps}\}$), whose dimensions are 1, 2, 4, and 13, respectively. Among them, predicting rest is a binary classification problem, producing rnp and cit belongs to the multi-class classification problem, while the generation of cps is a multi-label classification problem. Consequently, the predictions of rest, rnp, and cit are optimized using binary CE (BCE) and CE loss functions

$$\begin{aligned} L_{\text{rest}} &= \text{BCE}\left(\text{sigmoid}\left(\text{output}_{\text{rest}}\right)\right) \\ L_k &= \text{CE}\left(\text{softmax}\left(\text{output}_k\right)\right), \quad k \in \{\text{rnp}, \text{cit}\}. \end{aligned} \tag{6}$$

Generally, each output dimension is regarded as an independent Bernoulli distribution (or binary classification) in the multilabel classification problem. This approach is built on the premise that there is no interdependence among labels. However, due to the regular chord interval, the occurrences of pitches in a chord are not independent of each other. Therefore, we get rid of the conventional BCE loss function when predicting cps.

We use a new optimization scheme inspired by Sun et al. [36], where Sun et al. [36] proposed a unified formula (1) of classification loss functions (e.g., softmax CE loss [37], [38]) and metric loss functions (e.g., triplet loss [39], [40]) to maximize within-class similarity scores s_{pos} and minimize between-class similarity scores s_{neg}. This loss function iterates through every similarity pair to reduce ($s_{\text{neg}} - s_{\text{pos}}$). We remove the scale factor γ and the margin m from the formula, which are unnecessary in this article.

For the multilabel classification of cps, the within-class similarity scores correspond to the pitch classes that constitute the chord, while the between-class similarity scores correspond to the non-chord pitches. The last dimension of cps vector is set to 1 for triads to keep the number of predicted labels fixed since both the triads and sevenths are included in this article. Furthermore, we hope that the within-class similarity scores are as large as possible and the between-class similarity scores are as small as possible. Thus, two additional terms are introduced to form the final loss function of cps

$$\begin{aligned} L_{\text{cps}} &= \log\left[1 + \sum_{i=1}^{K}\sum_{j=1}^{L}\exp\left(s_{\text{neg}}^{j} - s_{\text{pos}}^{i}\right)\right. \\ &\qquad \left. + \sum_{i=1}^{K}\exp\left(-s_{\text{pos}}^{i}\right) + \sum_{j=1}^{L}\exp\left(s_{\text{neg}}^{j}\right)\right] \\ &= \log\left[1 + \sum_{i=1}^{K}\exp\left(-s_{\text{pos}}^{i}\right)\right] + \log\left[1 + \sum_{j=1}^{L}\exp\left(s_{\text{neg}}^{j}\right)\right] \end{aligned} \tag{7}$$

where the similarity scores $s_{\text{pos}}^{i}(i = 1, 2, \ldots, K)$ and $s_{\text{neg}}^{j}(j = 1, 2, \ldots, L)$ are obtained by performing softmax on the output

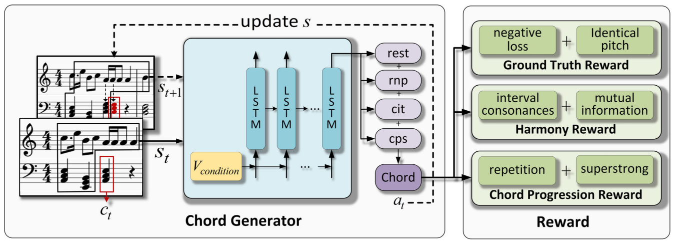


Fig. 3. Framework of the proposed RL-Chord, where the chord generator adopts the CLSTM model and the reward contains three modules. The state s is updated once generating a chord, and the reward r_t is calculated for this generated chord.

vector, i.e., $\text{output}_{\text{cps}}$. Given a chord sample, K and L are the numbers of within-class and between-class similarity scores, respectively ($K = 4$ and $L = 9$ in this article).

Ultimately, the generated chord at each step is learned by jointly optimizing four loss functions

$$L_{\text{CLSTM}} = L_{\text{rest}} + L_{\text{rnp}} + L_{\text{cit}} + L_{\text{cps}}. \tag{8}$$

IV. RL-Chord for Chord Generation

The RL-Chord uses RL to generate appropriate chords for a given melody. It consists of two parts, chord generator and reward, as shown in Fig. 3. The underlying architecture of the chord generator is the CLSTM proposed in Section III-C, and the instant reward includes three modules: ground-truth reward based on existing datasets, harmony reward based on music theory and mutual information, and chord progression reward based on several manual rules. Each reward module contains two sub-rewards, which will be elaborated on next.

We compare three categories of the model-free RL methods when implementing RL-Chord, i.e., policy-based method, value-based method, and the AC framework which combines the advantages of the policy-based and value-based methods. Specifically, the policy-based PG method uses the REINFORCE algorithm [41], the value-based DQN takes all the advantages of Double DQN [42], Dueling DQN [43], and NoisyNet [44], and the AC framework adopts the PPO algorithm [45] with generalized advantage estimator (GAE) [46]. We will explain how to apply these algorithms to the chord generation task in the follow-up.

A. Reward Definition

1) Ground-Truth Reward: The ground-truth reward assesses the generated chord with the help of the existing datasets.

a) Negative loss: Loss functions are used to measure the gap between the model output and the ground truth. Ideally, the loss will gradually decrease as the model continues learning. On the contrary, rewards are expected to increase as the training is going on. We calculate the negative maximum likelihood (ML) loss between the generated chord and the authentic target as a reward

$$r_{\text{neg\_loss}} = -L_{\text{ML}}. \tag{9}$$

b) Identical pitch: As a more intuitive reward, the identical pitch reward is the number of the shared pitch class of

TABLE II
SCORE OF INTERVAL CONSONANCE

Interval classification		Interval	Interval difference	Score
Conso-nance	extremely perfect	Perfect Unison Perfect Octave	0	10
	perfect	Perfect Fourth Perfect Fifth	5, 7	8
	imperfect	Major Third Minor Sixth	4, 8	6
		Minor Third Major Sixth	3, 9	5
Dissonance		Major Second Minor Seventh	2, 10	3
		Minor Second Major Seventh	1, 11	1
		Augmented Fourth Diminished Fifth	6	0

the generated and the target chord, i.e., the module of the intersection of two pitch sets

$$r_{\text{iden\_pitch}} = \left|\mathcal{C}_t^{\text{gen}} \cap \mathcal{C}_t^{\text{gt}}\right| \tag{10}$$

where $\mathcal{C}_t^{\text{gen}}$ and $\mathcal{C}_t^{\text{gt}}$ represent the generated and the target chord pitch set at the tth step, respectively.

Under the help of only ground truth, it may train a model over adapting to the training set, resulting in the model losing the ability to explore other possibilities. We will propose two other reward modules next.

2) Harmony Reward: The harmony reward evaluates the harmonicity between the generated chords and the given melodies. We draw support from the music interval consonances [47] and mutual information theory [48] to calculate two sub-rewards.

a) Interval consonance: According to auditory perception, the music interval can be divided into consonance and dissonance. The consonant interval sounds pleasant and relaxing, including extremely perfect consonance, perfect consonance, and imperfect consonance. While the dissonant interval sounds harsh and disgusting. We score different intervals according to the degree of consonance, and the scoring results are shown in Table II.

We use the melody segment of length 3, i.e., the melody notes at times $t-1$, t, and $t+1$, to calculate the interval consonances reward. Specifically, we calculate the weighted sum of the consonance scores of each melody note and its corresponding chord, where the consonance scores are obtained from Table II according to the interval difference, and the weight is the duration of each melody note. The weighted sum is divided by the sum of the melody note durations for normalization, and the calculation formula is as follows:

$$r_{\text{consonance}} = \frac{\sum_{i=t-1}^{t+1} m_i^d \cdot \sum_{j=1}^{N} \text{Score}(\text{Interval}(m_i, c_j))}{\sum_{i=t-1}^{t+1} m_i^d} \tag{11}$$

where m_i^d is the duration of melody note at time t, N is the number of notes forming the chord, $\text{Interval}(\cdot)$ is the interval difference, and $\text{Score}(\cdot)$ is the consonance score.

b) Mutual information: Mutual information is used for measuring the interdependence degree between two variables in the probability theory and information theory. For two random variables X and Y, their mutual information is the difference between the entropy of X before and after knowing Y, namely, $I(X, Y) = H(X) - H(X|Y)$. Previous studies applied mutual information to the generation tasks [14], [15], [21] to measure the consistency of the generated contents. Yi et al. [14] exploited mutual information to evaluate the consistency between adjacent lines in a poem. Nan et al. [21] used mutual information to calculate the amount of shared information between different voice parts.

In this article, we exploit mutual information to measure the consistency between melody and its chord accompaniment. Inspired by Nan et al. [21], we assume that melody M and chord C are two random variables, along with $M_i = \{m_t^i\}_{t=0}^T$ and $C_i = \{c_t^i\}_{t=0}^T$ are the ith samples in the dataset, and the mutual information between them is calculated as follows:

$$\begin{aligned} I(M, C) &= \sum_{M,C} p(M, C) \log \frac{p(M, C)}{p(M)p(C)} \\ &\approx \sum_{M_i, C_i} \sum_{t=0}^{T} \left[\log p\left(c_t^i \middle| M_i, c_{<t}^i\right) - \log p\left(c_t^i \middle| c_{<t}^i\right)\right]. \end{aligned} \tag{12}$$

Consequently, the mutual information reward is defined as

$$\begin{aligned} r_{\text{mutual\_info}} = \log p\left(c_t^i \middle| m_{t-L/2:t+L/2-2}^i, c_{<t}^i\right) \\ - \log p\left(c_t^i \middle| c_{<t}^i\right) \end{aligned} \tag{13}$$

where $c_{<t}^i$ are the chords generated before t. Two MLE models need to be pretrained to learn $\log p(c_t^i | m_{t-L/2:t+L/2-2}^i, c_{<t}^i)$ and $\log p(c_t^i | c_{<t}^i)$, respectively. For the former, we directly use the CLSTM model, while the latter could be a simple two-layer LSTM.

3) Chord Progression Reward: The chord progression reward is proposed to evaluate the chord transitions. On one hand, we should avoid repeating chords that last long. On the other hand, chord transitions should not be too sudden or aggressive.

a) Repetition: Chord repetition is allowed, but too much repetition will affect the quality of the generated chord progression. In extreme cases, there is only one kind of chord in the whole chord progression. On account of the one-to-one correspondence between chords and notes, the repetition of chords is common in the represented chord sequence. However, the onset of a bar often marks a new chord. The cross-bar chord repetition occurs when the chord at the onset of a bar is the same as that at the previous step, which we hope to avoid but not prohibit. The definition of the repetition reward is

$$r_{\text{repetition}}(c_t | c_{t-1}) = \begin{cases} -1, & \text{if } c_t = c_{t-1} \text{ and } t = \text{bar}_0 \\ 1, & \text{if } c_t = c_{t-1} \text{ and } t \neq \text{bar}_0 \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

where bar_0 indicates the bar onset. If the chord repeats at bar_0, penalize -1; repeats within a bar, reward 1.

b) Superstrong: As mentioned in [49], the superstrong progression occurs when the root of the second chord is a second step up or down. For instance, the current chord pitch set is $\{1, 3, 5\}$, and the previous one is $\{2, 4, 6\}$. We give a weak penalty to this kind of chord that does not appear often, i.e., when the pitch intersection set of the current and previous chord is empty, penalize 1

$$r_{\text{superstrong}}(c_t|c_{t-1}) = \begin{cases} -1, & \text{if } c_t \cap c_{t-1} = \varnothing \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

We do not prohibit the occurrence of superstrong progression as well. It is likely that when adjacent chord pitch sets have the same pitch class, the chord transition will sound more fluent, or else it will be relatively abrupt.

The instant reward r_t for the action to be taken at time t is

$$r_t = \alpha r_t^{\text{ground\_truth}} + \beta r_t^{\text{harmony}} + \gamma r_t^{\text{chord\_progression}} \tag{16}$$

where α, β, and γ are weight coefficients controlling the importance of each reward module in guiding chord generation.

B. Combining RL With CLSTM

1) Policy Gradient: We apply the Monte-Carlo-based PG algorithm (i.e., REINFORCE algorithm [41]) on CLSTM to construct PG-Chord. PG-Chord aims to train an agent that maximizes the expected returns. The network is updated once after generating a complete chord sequence. The reward at each step r_t is obtained as the training goes and the returns G_t are calculated at the end of the episode. Finally, the chord generator will be optimized according to the returns. The loss function is as follows:

$$L(\theta) = -\mathbb{E}\left[\sum_{t=0}^{T}\left(\sum_{l=t}^{T}\gamma^{l-t}r_l - b\right)\nabla \log \pi_\theta(a_t|s_t)\right] \tag{17}$$

where $0 < \gamma < 1$ is the discount factorand, b is the baseline computed by averaging the returns for accelerating the learning. $G_t = \sum_{l=t}^{T}\gamma^{l-t}r_l$ is the discounted return at the tth step. Since we divide the chord into four parts, a_t is composed of four subactions. Thus, the agent π_θ in (17) is further expanded into

$$\pi_\theta(a_t|s_t) = \prod_{j\in[\text{rest,rnp,cit,cps}]} \pi_\theta^j\left(a_t^j\middle|s_t\right) \tag{18}$$

where π_θ^{rest}, π_θ^{rnp}, π_θ^{cit}, and π_θ^{cps} share the same agent network, and different parts of the chord are generated through four fully connected layers.

2) Deep Q-Network: In this section, the DQN combined with multiple tricks is used to construct DQN-Chord on CLSTM. DQN-Chord aims to approximate an optimal Q value function using neural networks. The conventional DQN adopts the off-policy temporal differential (TD) learning strategy and usually configures an experience replay buffer. For the chord generation task in this article, we prefer to learn the structure and transition rules of the whole chord progression than learning the single transfer (s_t, a_t, r_t, s_{t+1}). Therefore, we discard the experience pool and update the model parameters after generating a complete chord sequence.

We first take advantage of Double DQN [42] to eliminate the problem of overestimation in common DQN, which uses two Q-networks to select the action and calculate the target Q value, respectively

$$L(\theta) = \mathbb{E}\left[\sum_{t=0}^{T}(r_t + \gamma \max_a Q'_{\theta'}(s_{t+1}, a) - Q_\theta(s_t, a_t))^2\right] \tag{19}$$

where Q is the prediction Q network and Q' is the target Q network. Next, we adopt Dueling DQN [43], which separates the Q network into a value function $V(s)$ and an advantage function $A(s, a)$. Finally, we implement NoisyNet [44] to promote the exploration of action space. As mentioned before, the action here still consists of four parts, and Q and Q' in (19) can be further expanded to

$$\begin{aligned} \max_a\ Q'_{\theta'}(s_{t+1}, a) &= \sum_{j\in[\text{rest,rnp,cit,cps}]} \max_{a^j}\ Q'_{\theta'}(s_{t+1}, a^j) \\ Q_\theta(s_t, a_t) &= \sum_{j\in[\text{rest,rnp,cit,cps}]} Q_\theta(s_t, a_t^j). \end{aligned} \tag{20}$$

3) Actor-Critic: The PPO with GAE is used to construct AC-Chord on CLSTM. AC-Chord includes an actor for generating chords and a critic for estimating the expectation of future returns.

As the baseline model of OpenAI, PPO has achieved good performance in many tasks, but it has not been applied to music generation. In this article, Clipped Surrogate Objective [45] is selected to solve the unconstrained optimization problem of TRPO [50]. The objective function is as follows:

$$\begin{aligned} &L^{\text{CLIP}}(\theta) = \mathbb{E}\left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\right] \\ &\text{s.t. } r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \end{aligned} \tag{21}$$

where $0 < \epsilon < 1$, and $\hat{A}_t$ is the discounted advantage function of GAE [46], which is regarded as a solution to the tradeoff between bias and variance under the AC framework.

The critic produces value function $V_\omega(s_t)$ by feeding s_t into several fully connected layers and activation layers. The parameters of the critic are updated by minimizing the mean squared error (MSE) between $V_\omega(s_t)$ and cumulative discounted reward G_t. In addition, we use the entropy-based exploration mechanism to limit the output distribution of the policy function, so as to better explore the environment. The loss functions of AC-Chord are as follows:

$$\begin{aligned} L_{\text{actor}} &= -\sum_{j\in[\text{rest,rnp,cit,cps}]} L^{\text{CLIP}}\left(\theta, a_t^j\right) \\ L_{\text{critic}} &= \mathbb{E}\left[\sum_{t=0}^{T}\|V_\omega(s_t) - G_t\|^2\right] \\ L_{\text{entropy}} &= \sum_{j\in[\text{rest,rnp,cit,cps}]}\sum_{t=0}^{T}\sum_{a}\pi_\theta(a^j|s_t)\log\pi_\theta(a^j|s_t) \\ L_{\text{AC-Chord}} &= L_{\text{actor}} + \alpha_1 L_{\text{critic}} + \alpha_2 L_{\text{entropy}} \end{aligned} \tag{22}$$

where α_1, α_2 are the loss weights.

Optimizing only the loss function of the RL algorithm brings much uncertainty and instability when training the model. Due to the lack of supervised signals, the model gets lost easily and leads to the explosive increase in MLE loss. To mitigate this problem, we adopt the way of jointly training CLSTM loss and RL loss

$$L_{\text{RL}-\text{Chord}} = \beta L_{\text{CLSTM}} + (1-\beta) L_{\text{RL}} \tag{23}$$

where $0 < \beta < 1$ is a variable weight that is slightly different for distinct RL algorithms.

C. Zero-Shot Chord Generation for CF Melodies

The lack of melody–chord paired datasets limits the potential of chord generation models in low-resource scenarios. In this section, we attempt to harmonize CF melodies without any CF chord data. Fortunately, we collect plenty of CF melodies. As mentioned earlier, melody and chord share some information, and generally, the chord containing melody notes is more likely to be a harmonious one. Hence, we wonder whether we could use only melody data to train a "style" classifier which can distinguish different styles of melodies and chord progressions after training. Here the "style" denotes distinct datasets.

Two existing melody–chord paired datasets, NMD and Wiki, are exploited to verify the above idea, and the datasets will be elaborated on in the next section. Specifically, we construct a style classifier, which is a two-layer LSTM with attention mechanism followed by a CNN block and a linear layer. In the training stage, the classifier takes the sequences of melody pitch and duration as input to classify the melodies. In the inference stage, the sequences of chord root pitch and duration are fed into the classifier to categorize chord accompaniments. We reduce the dimension of the final output features and visualize them using t-SNE, and the visualization result is shown in Fig. 4. It can be seen that the classifier can distinguish different styles of melodies very well, while the chord classification results are not good enough but still acceptable. In addition, the distributions of melodies and chord accompaniments overlap each other for the same style, while different styles are distinguishable.

In view of the promising results, we train a classifier with the melody data of Wiki and CF, hoping that the classifier can identify the chord progressions in CF style. Table III presents the results of classification accuracy. For the input $x \in \{\text{melody}, \text{chord}\}$ of dataset $d \in \{\text{CF}, \text{Wiki}\}$, the classification accuracy acc_d^x is the number of correctly classified samples divided by the total number of x in d. As we can see, the Wiki-CF classifier distinguishes different styles of melodies very well, and the classification accuracy of chords in Wiki style is not bad. After that, we pretrain a DQN-Chord model using the Wiki dataset. Then the Wiki-CF classifier with fixed parameters is attached to the pretrained DQN-Chord to calculate the classification CE loss for the generated chord progressions given the melody inputs in CF style. The classification loss and RL loss are joined to fine-tune the parameters of DQN-Chord to make the generated chords possess CF style, and the fine-tuned model is called CF-Chord. Fig. 5 shows the framework of CF-Chord, and the loss function is as follows:

$$L_{\text{CF}-\text{Chord}} = \beta L_{\text{Classify}} + (1-\beta) L_{\text{RL}} \tag{24}$$

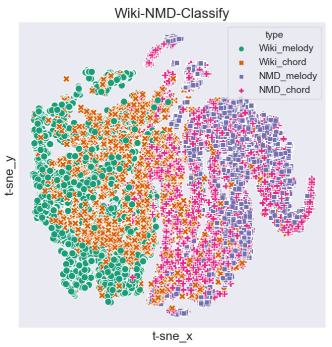


Fig. 4. t-SNE visualization of melody and chord features learned by Wiki-NMD classifier.

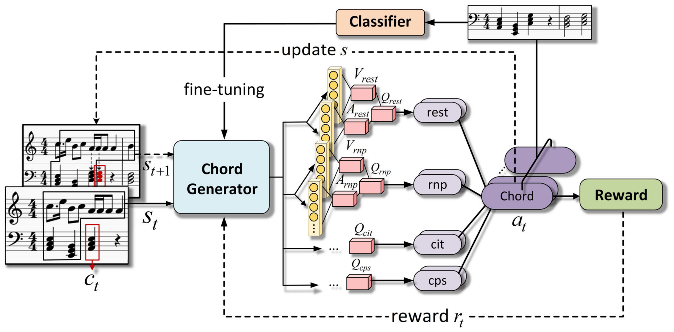


Fig. 5. Framework of zero-shot chord generation for CF melodies. The model architecture is the same as the DQN-Chord. Once completing the chord progression, the classifier with fixed parameters is used to fine-tune the pretrained DQN-Chord for generating appropriate chords in CF style.

TABLE III
CLASSIFICATION RESULTS OF THE WIKI-CF CLASSIFIER

Data set	Accuracy	
	melody	chord
CF	0.9750	\
Wiki	0.9780	0.8990

where β is more inclined to the classification loss. It is noteworthy that we remove the ground-truth reward due to the lack of CF chord data.

V. EXPERIMENTS

A. Experimental Setup

1) Datasets: Three datasets are introduced in this article, two of which are used to train the melody harmonization models, and one is used for zero-shot CF chord generation. The first one is the Nottingham Music Dataset[1] (NMD) that contains more than 1000 British and American folk songs, each including melody and corresponding chord accompaniment. The second dataset (Wiki) comes from a public lead sheet repository[2] provided by Lim et al. [11]. They collected 2252 lead sheets, each containing one chord per bar. The Wiki dataset covers multiple music genres, such as rock, pop, and jazz. In addition, we collected a CF melody dataset containing more than 3000 music pieces. Specifically, we manually digitized the musical notations into MIDI format from the

[1] abc.sourceforge.net/NMD/
[2] http://marg.snu.ac.kr/chord_generation/

TABLE IV
Dataset Statistics

Data set	Number of chord types	Number of chords per bar	Genres	Number of 64-length snippets	Number of 128-length snippets
NMD	116	$\leqslant 4$	$= 1$	112,385	62,319
Wiki	310	$\leqslant 1$	> 1	221,870	97,070

record of *Chinese Folk Music Integration*[3] using the software MuseScore.[4]

To simplify the problem, we are not going to cover all chord types, but only consider triad chords and seventh chords that frequently appear in the NMD and Wiki datasets. More specifically, the triads include different inversion types of major, minor, augmented, and diminished chords, and the seventh chords contain different inversion types of major, dominant, minor–major, minor, and half-diminished chords. For both the datasets, we delete the pieces that do not contain chord and convert the chords that are not triads or sevenths into triads or sevenths through the best matching principle.[5] In addition, we split chords and align the notes and chords in the light of the proposed chord representation. Each piece is transposed to all the root notes for data augmentation by shifting up five semi-tones and down six semi-tones, guaranteeing that the pitch range after transposing does not exceed that of original music. Finally, we divide the processed music into music segments of the same length (64 or 128) for model training. The statistics of the datasets are shown in Table IV.

2) Implementation Details: Each dataset is randomly divided into 80% training set, 10% validation set, and 10% test set. All the models are trained with the SGD optimizer. The learning rate is 0.001, the batch size is 64, and L2 regularization is used to reduce the overfitting. The length L of the condition window is set to 8. The number of hidden nodes of LSTM in all the models is 512. The discount factor γ of RL-Chord is 0.99, and the parameter λ of GAE is 0.95. The loss weights β of AC-Chord and DQN-Chord are 0.3, while the loss weight of PG-Chord increases linearly from 0 to 0.3 through 30 epochs and then fixed. The loss weight of CF-Chord is 0.7. RL-Chord consists of two training stages. First, two MLE models for mutual information reward and the classifier for CF chord generation are trained, and then the parameters of these models are fixed to train the RL-Chord model. All RL-Chords are combined with the Teacher Forcing [51] to guide the learning of chord generators.

B. Experimental Results

1) Automatic Evaluation:

a) Music metrics: To evaluate the generated chords quantitatively, we use three music metrics defined by Yeh et al. [9], i.e., chord tonal distance (CTD), pitch consonance score (PCS), and melody—chord tonal distance (MCTD). Among

[3]One of the major national cultural project led by the former Ministry of Culture, National Ethnic Affairs Commission and Chinese Musicians Association from 1984 to 2001. The website is http://cefla.org.cn/project/book

[4]https://musescore.org

[5]Convert the chord into a triad or a seventh chord whose interval relationships are closest to its.

TABLE V
Comparison Results Between BLSTM and CLSTM With One-Hot Representation

Data set	Model	CHS	CNR	CTD	DC	PCS	MCTD
NMD	Ground truth	1.0000	0.3449	0.9296	3.8820	1.8516	0.0394
	BLSTM	0.8214	0.4115	0.5646	**13.328**	**0.3336**	**0.0510**
	CLSTM	**0.8353**	**0.3131**	**0.5959**	15.6960	0.1605	0.0600
Wiki	Ground truth	1.0000	0.3285	0.8536	7.7040	1.5095	0.0417
	BLSTM	0.7706	0.4407	0.4833	19.650	**0.4576**	**0.0504**
	CLSTM	**0.8026**	**0.3306**	**0.5166**	**13.7540**	0.4440	0.0521

them, CTD measures the closeness of two adjacent chords, and PCS and MCTD measure the harmonicity between melody and chord. Apart from the above metrics, three additional metrics are proposed in this article, namely, chord histogram similarity (CHS), chord-to-note ratio (CNR), and discordant chord (DC).

1) *CHS:* The Euclidean distance similarity between the histograms of generated chord and the human-composed chord under the same melody condition. The higher the CHS, the closer the distribution of generated chord is to the ground truth.
2) *CNR:* The ratio of the number of chords to the number of melody notes in a piece of music. The larger the CNR, the denser the chords.
3) *DC:* The number of discordant chords in a piece of music. As mentioned in [52] that if some melody notes are a part of the chord pitch set, the chord is more likely to accompany the melody harmoniously. So the discordant chord is defined as the chord whose pitch set does not contain any melody note of the bar where it is located. The higher the DC, the more disharmonious the generated chord progression.

The generated result is generally better when the metrics values are closer to that of the ground truth.

b) Comparative study: Under the OH representation, we first compare the BLSTM-based baseline and the proposed CLSTM. Then we use CLSTM to compare OH representation and our proposed MH representation. After that, two categories of methods are compared. One is based on MLE, which includes the BLSTM and CLSTM. The other is the RL model which contains three RL algorithms, i.e., PG-Chord, DQN-Chord, and AC-Chord. The BLSTM-based baseline is a two-layer BLSTM [11], which only takes the melody in the condition window as the input to predict the corresponding chord, without considering the previously generated chords.

The comparison results between BLSTM and CLSTM are shown in Table V. On both the datasets, the generated results of CLSTM are closer to the ground truth in terms of chord type, quantity, and chord transitions (CHS, CNR, and CTD), while the chord progressions generated by BLSTM are denser and lack fluctuations (higher CNR, lower CTD). As the variety of chords in the dataset increases, CLSTM produces fewer dissonant chords (lower DC). However, CLSTM performs slightly worse than BLSTM on metrics evaluating harmonicity (PCS, MCTD). We speculate that BLSTM mainly focuses on learning the correspondence between melodies and chords, while CLSTM needs to learn more information like chord

TABLE VI

COMPARISON RESULTS BETWEEN REPRESENTATIONS OF OH AND MH USING CLSTM

Data set	Representation	CHS	CNR	CTD	DC	PCS	MCTD
NMD	Ground truth	1.0000	0.3449	0.9296	3.8820	1.8516	0.0394
	One-hot	**0.8353**	0.3131	0.5959	15.6960	0.1605	0.0600
	Multi-hot	0.7884	**0.3718**	**0.6645**	**7.6860**	**1.6282**	**0.0424**
Wiki	Ground truth	1.0000	0.3285	0.8536	7.7040	1.5095	0.0417
	One-hot	**0.8026**	**0.3306**	0.5166	13.7540	0.4440	0.0521
	Multi-hot	0.7617	0.3319	**0.5601**	**8.4840**	**1.4485**	**0.0423**

TABLE VII

OBJECTIVE EVALUATION SCORES OF DIFFERENT MODELS. UNLESS NOTED, CHORDS ARE REPRESENTED WITH MH

Data set	Model	CHS	CNR	CTD	DC	PCS	MCTD
NMD	Ground truth	1.0000	0.3449	0.9296	3.8820	1.8516	0.0394
	BLSTM w/ OH	0.8214	0.4115	0.5646	13.328	0.3336	0.0510
	CLSTM	0.7884	0.3718	0.6645	7.6860	1.6282	0.0424
	PG-Chord	0.8002	0.3641	0.6452	8.3840	1.6368	0.0416
	DQN-Chord	**0.8337**	**0.3546**	**0.6900**	5.3200	**1.8127**	0.0405
	AC-Chord	0.8108	0.3708	0.6439	**5.3160**	1.7400	**0.0403**
Wiki	Ground truth	1.0000	0.3285	0.8536	7.7040	1.5133	0.0417
	BLSTM w/ OH	0.7706	0.4407	0.4833	19.650	0.4576	0.0504
	CLSTM	0.7617	0.3319	0.5601	8.4840	1.5192	0.0423
	PG-Chord	0.7684	**0.3308**	0.5149	9.4540	1.3243	0.0445
	DQN-Chord	**0.7803**	0.3322	**0.7359**	**7.9380**	1.6705	**0.0419**
	AC-Chord	0.7691	0.3348	0.6206	8.4000	**1.5081**	0.0430

TABLE VIII

MODEL COMPARISON RESULTS WITH DIFFERENT SEQUENCE LENGTHS OF MUSIC ON THE WIKI DATASET

Seq length	Model	CHS	CNR	CTD	DC	PCS	MCTD
64	Ground Truth	1.0000	0.3285	0.8536	7.7040	1.5133	0.0417
	CLSTM	0.7617	**0.3319**	0.5601	8.4840	**1.5192**	0.0423
	DQN-Chord	**0.7803**	0.3322	**0.7359**	**7.9380**	1.6705	**0.0419**
128	Ground Truth	1.0000	0.2628	0.8645	8.3140	1.4016	0.0488
	CLSTM	**0.7856**	0.2686	0.3336	13.744	0.8653	0.0580
	DQN-Chord	0.7848	**0.2662**	**0.5955**	**11.308**	**1.3246**	**0.0523**

transitions. Fortunately, CLSTM combined with the MH representation will remedy this problem (see Table VI). But MH cannot help BLSTM solve its defects on CNR and CTD.

Table VI demonstrates the comparison results of two representation methods. The dimension of the OH increases as the number of chord types expands. The MH can represent all the triads and seventh chords by a 20-D vector. We conclude from the table that the chords generated with MH are closer to the ground truth on both the datasets. In particular, the chords generated by the OH perform poorly on CTD, DC, PCS, and MCTD, implying inferior harmonicity between the melody and chord progression as well as chord transition without many fluctuations. Therefore, we adopt the MH by default unless otherwise specified in the follow-up.

Table VII shows comparison of the objective evaluation scores of different methods. On the whole, the generated results of DQN-Chord are closest to the ground truth on both the datasets, while the baseline BLSTM w/ OH possesses the worst performance. For the NMD dataset, the CHS score and the CNR score of the DQN-Chord are closest to the ground truth, indicating the generated chords are similar to the human-composed in terms of the type and number, which also proves that the model learns the distribution of dataset well. CLSTM and PG-Chord score higher on DC, suggesting they generate more discordant chords. Both DQN-Chord and AC-Chord score high on PCS and low on MCTD, demonstrating good consistency and harmonicity between the generated chord and melody. Instead, CLSTM scores low on PCS and high on MCTD. The similar scores of DQN-Chord and AC-Chord imply their comparable results. As for the Wiki dataset, the results of BLSTM w/ OH are still far from the ground truth, especially with a large number of discordant chords (the highest DC). The low PCS score and high MCTD score of PG-Chord indicate the insufficient harmonicity between chord and melody. AC-Chord still obtains decent evaluation scores close to that of DQN-Chord, but it is inferior to DQN-Chord on CTD and DC, which indicates that DQN-Chord performs better in learning chord transition rules.

In particular, we observed that the CNR score of BLSTM w/ OH is quite different from those of other models, which is always higher than the ground truth. This phenomenon shows that the generated chord progressions are relatively dense. In the NMD and Wiki datasets, it is common that there are one or two chords per bar. However, the number of generated chords per bar by BLSTM w/ OH is even the same as the number of melody notes sometimes. We attribute this phenomenon to data representation and model construction: chord generation at each step by BLSTM only depends on melody information and does not consider the previous chords, leading to poor learning of chord transition. Meanwhile, the one-to-one correspondence of melody notes and chords cannot indicate the chord duration explicitly.

Reinforcement learning aims to maximize future returns, so it can theoretically generate music with long-term consistency. While the music generated by RNN models usually has quite a few repetitions and loses musical audibility after exceeding a certain length. To this end, we clip the pieces of music in the Wiki dataset into different lengths (64 and 128) and train the CLSTM and DQN-Chord models with music data of two lengths. The experimental results are shown in Table VIII. For the sequence length of 64, the distinctions between CLSTM and DQN-Chord on CHS, CNR, and MCTD are not obvious. The CLSTM gets a lower score on CTD, indicating that the chord transitions are probably too smooth without many fluctuations. DQN-Chord scores lower on DC and higher on PCS, demonstrating the decent harmonicity between the generated chord and melody. When the sequence length increases to 128, the superiority of the RL algorithm becomes more prominent. Especially for the metrics used to judge the harmonicity between melody and chord and the fluency of chord transition (CTD, DC, PCS, and MCTD), DQN-Chord shows better results. As for the statistical metrics used to evaluate the number and type of chords (CHS and CNR), CLSTM and DQN-Chord are comparable. But these metrics

TABLE IX

Ablation Study for DQN-Chord. r_{GT} Is the Ground-Truth Reward, r_H Is the Harmony Reward, and r_{CP} Is the Chord Progression Reward

Dataset	Model	CHS	CNR	CTD	DC	PCS	MCTD
NMD	Ground Truth	1.0000	0.3449	0.9296	3.8820	1.8516	0.0394
	DQN-Chord	0.8337	0.3546	0.6900	**5.3200**	**1.8127**	0.0405
	w/o r_{GT}	0.8377	0.3678	**0.8019**	5.7180	1.7926	**0.0400**
	w/o r_H	**0.8462**	**0.3527**	0.7371	6.4160	1.7110	0.0411
	w/o r_{CP}	0.8331	0.3617	0.7533	5.7220	1.7266	0.0408
Wiki	Ground Truth	1.0000	0.3285	0.8536	7.7040	1.5133	0.0417
	DQN-Chord	0.7803	0.3322	**0.7359**	**7.9380**	1.6705	**0.0419**
	w/o r_{GT}	0.7755	0.3333	0.6884	7.9580	**1.5976**	0.0420
	w/o r_H	0.7780	**0.3316**	0.6976	8.1160	1.6287	0.0420
	w/o r_{CP}	**0.7865**	0.3319	0.6847	8.1020	1.6783	0.0412

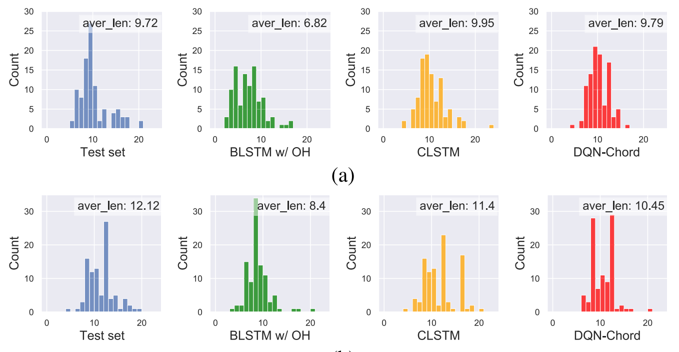


Fig. 6. Histogram of the length of the longest chord progression subsequence (in beats) copied from the training set. The *aver_len* is the mean length for the longest copied subsequences. (a) Results of plagiarism on the NMD dataset. (b) Results of plagiarism on the Wiki dataset.

cannot judge the quality of generated chords well and are more suitable for evaluating whether the model learns the distribution of datasets effectively.

c) Ablation study: We perform the ablation study for DQN-Chord by removing three reward modules, i.e., ground-truth reward, harmony reward, and chord progression reward. The results are illustrated in Table IX. We observed that no matter which reward module is eliminated, the impact on CHS and CNR is not significant. For the NMD dataset, removing the ground-truth reward brings chord transition (higher CTD) closer to the ground truth. However, it affects the harmonicity between chord and melody (higher DC, lower PCS). Removing the harmony reward or chord progression reward also destroys the harmonicity between melody and chord (worse DC, PCS, and MCTD). For the Wiki dataset, eliminating ground-truth reward makes PCS score closest to ground truth but brings CTD score far away from the ground truth. Except for the above effects, dropping the harmony reward increases the DC score additionally. Removing chord progression reward damages the harmonicity between melody and chord and the fluency of chord transition severely (worse CTD, DC, PCS, and MCTD). In principle, the combination of three reward modules still brings the best results. Among them, harmony reward contributes the most to the NMD dataset, and chord progression reward contributes the most to the Wiki dataset.

d) Plagiarism analysis: The above metrics only evaluate the similarity between the generated chord and ground truth, while overfitting the dataset is not what we want because it will ignore music creativity. Thus, we conduct a plagiarism analysis to evaluate the originality of generated chord progressions. Inspired by Hadjeres et al. [5], for a given chord accompaniment, we use the length of the longest chord progression subsequence that can be found in the training set as a measure of plagiarism. For each dataset, we randomly select 100 music clips from the test set, and then regenerate chords for the melodies with different methods, i.e., BLSTM w/ OH, CLSTM, and DQN-Chord.

For each method, we plot the histogram of the length of the longest copied chord progression subsequence, as shown in Fig. 6. For the NMD dataset, the distributions of the length of the longest plagiarized subsequence for CLSTM and DQN-Chord are similar, peaking around nine beats. And their mean lengths for the longest copied subsequence are close to that of the test set. For the Wiki dataset, chord progressions from the test set tend to be more "plagiaristic" than other methods, with a longer mean length for the longest copied subsequences. CLSTM plagiarizes more long subsequences (16 beats) from the training set. Surprisingly, the chords generated by BLSTM w/ OH are the least "plagiaristic" compared with other methods on both the datasets, and it seems that insufficient learning of the model can account for this as its performances in objective evaluation and the following listening test are poor.

2) Human Evaluation: As a product of creativity, music is subjective in nature, so people need to make a final judgment on the quality of the generated music. We conducted two listening tests. The first one compared distinct methods by asking subjects to score the generated chords. In the second task, the subjects were asked to rate the CF chords generated by the CF-Chord model to verify the effectiveness of the proposed unsupervised chord generation method. A total of 134 subjects participated in these two surveys containing 45 females and 89 males, and the age range is from 18 to 30 years old. Among them, about 20% of subjects have learning experiences of musical instruments or vocals and are familiar with harmonic theory more or less.

We compare human-composed chords with those generated by BLSTM w/ OH, CLSTM w/ OH, CLSTM (w/ MH), and DQN-Chord. For each method, we randomly selected ten music snippets with durations ranging from 15 to 40 s, resulting in a total of 50 music clips. Each subject was invited to listen to these 50 music clips and score the chord progressions based on the following four music metrics [9], [23].

1) *Coherence:* How well the chord progression matches with the melody in terms of harmonicity and phrasing.
2) *Chord Progression:* How reasonable or smooth the chord progression is on its own, independent of the melody.
3) *Interestingness:* How surprising or innovative the chord progression is on its own, independent of the melody.
4) *Overall:* How well the whole harmonization is in comprehensive consideration.

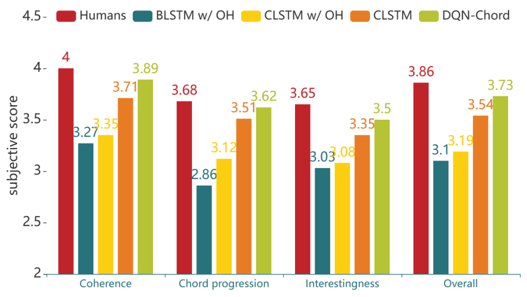


Fig. 7. Histogram of human evaluation results.

All the metrics are scored on a 5-point Likert scale where 1 represents the lowest score and 5 represents the highest score.

The human evaluation results are shown in Fig. 7. The chords created by humans achieve the best scores. Compared with BLSTM w/ OH, CLSTM w/ OH has obvious enhancement on Chord progression, but only slight improvement on other metrics. By comparing the results of CLSTM w/ OH and CLSTM (w/ MH), we conclude that MH outperforms OH on all subjective metrics. Among all four methods, DQN-Chord possesses the best results, especially the score on Chord progression is pretty close to that of ground truth. Compared with BLSTM w/ OH, CLSTM (w/ MH) has obvious improvement in terms of all the metrics. Combining DQN with CLSTM further improves the effect.

Through the Mann–Whitney U test, we conclude that the chords generated by OH and MH based on CLSTM have significant differences ($p < 0.05$) on all the subjective metrics. Adopting the OH, the generated results of CLSTM are significantly different from those of BLSTM only on Chord progression. The chords generated by BLSTM w/ OH and humans or CLSTM or DQN-Chord have significant differences on all the metrics. The chords generated by CLSTM are not significantly different from the human-composed ones only on Chord progression. There is no significant difference between ground truth and the chords generated by DQN-Chord on all the metrics. The chords generated by CLSTM and DQN-Chord have significant differences on Coherence and Overall.

To further confirm the effectiveness of the unsupervised CF chord generation method proposed in Section IV-C, we used the CF-Chord model to generate chords for ten CF melodies and then asked the 134 invited subjects to rate the generated chords according to the harmony/matching degree between the chord progression and melody. The chord matching degree is divided into five levels: "mismatch," "match a little," "not bad," "fair match," and "match well." The rating results are shown in Fig. 8. Most chord accompaniments are evaluated as "not bad," followed by "fair match." The numbers of "match a little" and "match well" are close, and the fewest chord accompaniments are evaluated as "mismatch." More than 3/4 of the chord accompaniments are considered not bad or better, nearly 50% are regarded as appropriate accompaniments ("match well" or "fair match"), and some of them even get high grades. We conclude that the CF-Chord model can generate satisfactory CF chord progressions. Several generation examples are displayed in Fig. 9.

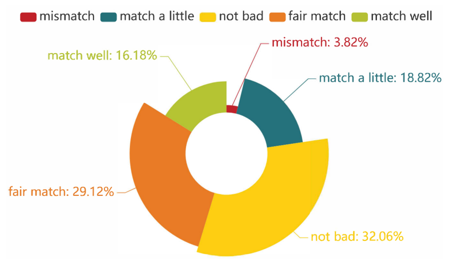


Fig. 8. Rating results of the generated CF chords.

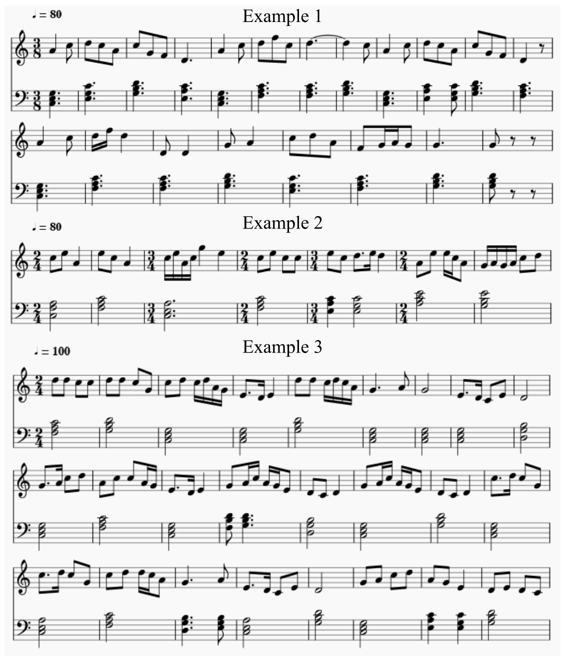
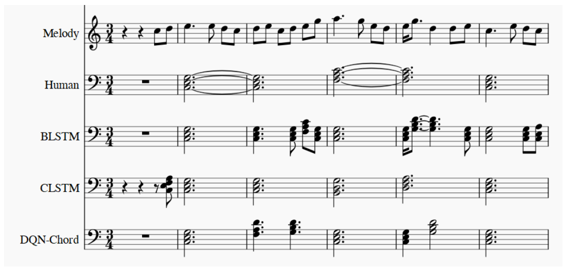
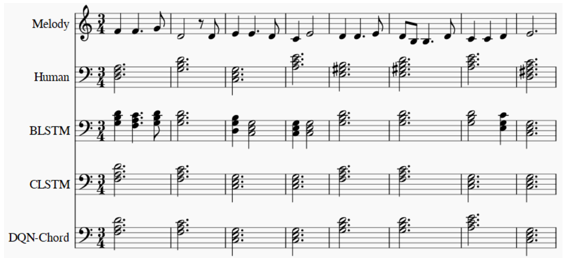


Fig. 9. Examples of chord progressions in CF style generated by CF-Chord.

3) Qualitative Evaluation: In this section, we analyze the chord progressions generated by different models qualitatively through music score visualization. Fig. 10 shows the examples of different models harmonizing two random melodies from the NMD and Wiki datasets. The melody in Fig. 10(a) is randomly selected from the NMD dataset. As we can see, BLSTM does not learn the chord durations effectively, resulting in relatively dense chord progression (even including a chord whose duration is the same as the 16th note). This phenomenon is very rare in the training set. The CLSTM produces a stable chord progression without too many fluctuations, and it learns the chord duration well compared with BLSTM. However, an unusual chord whose duration is the same as the 8th note appears at the end of the first pickup measure. The chords generated by DQN-Chord are more diverse, and the chord progression contains two chords per bar sometimes, bringing new creativity to chord creation. The melody in Fig. 10(b) comes from the Wiki dataset. We can see clearly that all the models can learn the distinctive characteristic that each bar has only one corresponding chord besides BLSTM. The chords generated by all the models are not as diverse as those created by humans, among which the generated chords of CLSTM differ from that of DQN-Chord only in 5–7 bars. But in these bars, more perfect unison/perfect octave intervals appear between the chords generated by DQN-Chord and melody



(a)



(b)

Fig. 10. Chord generation examples for qualitative evaluation. (a) Examples of chord progressions generated by humans, BLSTM, CLSTM, and DQN-Chord for a melody from the NMD dataset. (b) Examples of chord progressions generated by humans, BLSTM, CLSTM, and DQN-Chord for a melody from the Wiki dataset.

notes, making the chord progression more harmonious and natural. Nevertheless, there is still a gap between the chords generated by DQN-Chord and those created by humans.

VI. Conclusion

In this article, we proposed an RL-based melody harmonization system, RL-Chord, which generated better chord progressions than the compared models. The newly proposed representation represented a large number of chord types without increasing the representation dimension. The proposed CLSTM enhanced the ability of BLSTM in learning the chord progression by fully considering the context of the current chord. For the first time, three RL algorithms with well-designed reward modules (i.e., PG, Q-learning, and AC) were compared on the melody harmonization task. Furthermore, a style classifier was constructed to fine-tune the DQN-Chord to realize the unsupervised chord generation in CF style. The experimental results showed that CLSTM with the proposed representation had great improvement over the baseline model, and DQN-Chord achieved the best results in automatic evaluation, human evaluation, and qualitative evaluation.

This article focuses on the generation of column chords, i.e., all the pitches of the chord are played simultaneously. In the future, we will further explore broken chords (i.e., the chord pitches are played in a specific order) to make the generated accompaniments more vivid and creative. We also plan to realize online chord generation system to strengthen the interaction with human beings.

References

[1] J.-P. Briot, G. Hadjeres, and F.-D. Pachet, *Deep Learning Techniques for Music Generation*, vol. 1. New York, NY, USA: Springer, 2020.

[2] S. Ji, J. Luo, and X. Yang, "A comprehensive survey on deep music generation: Multi-level representations, algorithms, evaluations, and future directions," 2020, *arXiv:2011.06801*.

[3] E. Waite et al., "Generating long-term structure in songs and stories," *Web Blog Post. Magenta*, vol. 15, no. 4, Jul. 2016.

[4] A. Roberts, J. Engel, C. Raffel, C. Hawthorne, and D. Eck, "A hierarchical latent vector model for learning long-term structure in music," in *Proc. 35th Int. Conf. Mach. Learn. (ICML)*, 2018, pp. 4364–4373.

[5] G. Hadjeres, F. Pachet, and F. Nielsen, "DeepBach: A steerable model for bach chorales generation," in *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, 2017, pp. 1362–1371.

[6] J. Thickstun, Z. Harchaoui, D. P. Foster, and S. M. Kakade, "Coupled recurrent models for polyphonic music composition," in *Proc. 20th Int. Soc. Music Inf. Retr. Conf., ISMIR*, 2019, pp. 311–318.

[7] H.-W. Dong, W.-Y. Hsiao, L.-C. Yang, and Y.-H. Yang, "MuseGAN: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment," in *Proc. 32nd AAAI Conf. Artif. Intell.*, vol. 32, no. 1, 2018, pp. 34–41.

[8] Y. Z. Zhou, W. Chu, S. Young, and X. Chen, "BandNet: A neural network-based, multi-instrument beatles-style MIDI music composition machine," in *Proc. 20th Int. Soc. Music Inf. Retr. Conf. (ISMIR)*, 2019, pp. 655–662.

[9] Y.-C. Yeh et al., "Automatic melody harmonization with triad chords: A comparative study," *J. New Music Res.*, vol. 50, no. 1, pp. 37–51, Jan. 2021.

[10] H.-M. Liu and Y.-H. Yang, "Lead sheet generation and arrangement by conditional generative adversarial network," in *Proc. 17th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2018, pp. 722–727.

[11] H. Lim, S. Rhyu, and K. Lee, "Chord generation from symbolic melody using BLSTM networks," in *Proc. 18th Int. Soc. Music Inf. Retr. Conf. (ISMIR)*, 2017, pp. 621–627.

[12] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, "Sequence level training with recurrent neural networks," in *Proc. 4th Int. Conf. Learn. Represent. (ICLR)*, 2016, pp. 1–16.

[13] S. Wiseman and A. M. Rush, "Sequence-to-sequence learning as beam-search optimization," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 1296–1306.

[14] X. Yi, M. Sun, R. Li, and W. Li, "Automatic poetry generation with mutual reinforcement learning," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 3143–3153.

[15] M. Yang, W. Huang, W. Tu, Q. Qu, Y. Shen, and K. Lei, "Multitask learning and reinforcement learning for personalized dialog generation: An empirical study," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 49–62, Jan. 2021.

[16] Y. Keneshloo, T. Shi, N. Ramakrishnan, and C. K. Reddy, "Deep reinforcement learning for sequence-to-sequence models," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 7, pp. 2469–2489, Jul. 2019.

[17] Y. Ganin, T. Kulkarni, I. Babuschkin, S. A. Eslami, and O. Vinyals, "Synthesizing programs for images using reinforced adversarial learning," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2018, pp. 1666–1675.

[18] L. Yu, W. Zhang, J. Wang, and Y. Yu, "SeqGAN: Sequence generative adversarial nets with policy gradient," in *Proc. 31st AAAI Conf. Artif. Intell.*, vol. 31, no. 1, 2017, pp. 1–7.

[19] N. Jaques, S. Gu, R. E. Turner, and D. Eck, "Tuning recurrent neural networks with reinforcement learning," in *Proc. 5th Int. Conf. Learn. Represent. Workshop*, 2017, pp. 1–12. [Online]. Available: https://openreview.net/forum?id=Syyv2e-Kx

[20] N. Jiang, S. Jin, Z. Duan, and C. Zhang, "RL-Duet: Online music accompaniment generation using deep reinforcement learning," in *Proc. 34th AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 710–718.

[21] N. Jiang, S. Jin, Z. Duan, and C. Zhang, "When counterpoint meets Chinese folk melodies," in *Proc. Adv. Neural Inf. Proces. Syst.*, vol. 33, 2020, pp. 16258–16270.

[22] S. Shukla and H. Banka, "An automatic chord progression generator based on reinforcement learning," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2018, pp. 55–59.

[23] C.-E. Sun, Y.-W. Chen, H.-S. Lee, Y.-H. Chen, and H.-M. Wang, "Melody harmonization using orderless NADE, chord balancing, and blocked Gibbs sampling," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 4145–4149.

[24] R. Yang, D. Wang, Z. Wang, T. Chen, J. Jiang, and G. Xia, "Deep music analogy via latent representation disentanglement," in *Proc. 20th Int. Soc. Music Inf. Retr. Conf. (ISMIR)*, Delft, The Netherlands, Nov. 2019, pp. 596–603.

[25] J. Zhao and G. Xia, "Accomontage: Accompaniment arrangement via phrase selection and style transfer," in *Proc. 22nd Int. Soc. Music Inf. Retr. Conf. (ISMIR)*, Nov. 2021, pp. 833–840.

[26] N. Zhang, "Learning adversarial transformer for symbolic music generation," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jul. 2, 2020, doi: 10.1109/TNNLS.2020.2990746.

[27] Z. Wang et al., "PIANOTREE VAE: Structured representation learning for polyphonic music," in *Proc. 21st Int. Soc. Music Inf. Retr. Conf. (ISMIR)*, Montreal, QC, Canada, Oct. 2020, pp. 368–375.

[28] S. H. Hakimi, N. Bhonker, and R. El-Yaniv, "BEBOPNET: Deep neural models for personalized jazz improvisations," in *Proc. 21st Int. Soc. Music Inf. Retr. Conf. (ISMIR)*, Montreal, QC, Canada, Oct. 2020, pp. 828–836.

[29] W.-Y. Hsiao, J.-Y. Liu, Y.-C. Yeh, and Y.-H. Yang, "Compound word transformer: Learning to compose full-song music over dynamic directed hypergraphs," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, 2021, pp. 178–186.

[30] S. D. You and P.-S. Liu, "Automatic chord generation system using basic music theory and genetic algorithm," in *Proc. IEEE Int. Conf. Consum. Electron.-Taiwan (ICCE-TW)*, May 2016, pp. 1–2.

[31] I. Simon, D. Morris, and S. Basu, "MySong: Automatic accompaniment generation for vocal melodies," in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, Apr. 2008, pp. 725–734.

[32] H.-R. Lee and J.-S. Jang, "i-Ring: A system for humming transcription and chord generation," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, vol. 2, Jun. 2004, pp. 1031–1034.

[33] H. Tsushima, E. Nakamura, K. Itoyama, and K. Yoshii, "Interactive arrangement of chords and melodies based on a tree-structured generative model," in *Proc. 19th Int. Soc. Music Inf. Retr. Conf. (ISMIR)*, 2018, pp. 145–151.

[34] C.-Z. A. Huang, T. Cooijmans, A. Roberts, A. C. Courville, and D. Eck, "Counterpoint by convolution," in *Proc. 18th Int. Soc. Music Inf. Retr. Conf. (ISMIR)*, S. J. Cunningham, Z. Duan, X. Hu, and D. Turnbull, Eds., 2017, pp. 211–218.

[35] C. Jin, Y. Tie, Y. Bai, X. Lv, and S. Liu, "A style-specific music composition neural network," *Neural Process. Lett.*, vol. 52, no. 3, pp. 1893–1912, Dec. 2020.

[36] Y. Sun et al., "Circle loss: A unified perspective of pair similarity optimization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6398–6407.

[37] Y. Sun, X. Wang, and X. Tang, "Deep learning face representation from predicting 10,000 classes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1891–1898.

[38] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for convolutional neural networks," in *Proc. 37th Int. Conf. Machin. Learn. (ICML)*, vol. 2, no. 3, 2016, p. 7.

[39] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*. Cham, Switzerland: Springer, 2015, pp. 84–92.

[40] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 815–823.

[41] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, no. 3, pp. 229–256, 1992.

[42] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. 30th AAAI Conf. Artif. Intell.*, vol. 30, no. 1, 2016, pp. 1–7.

[43] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *Proc. 33rd Int. Conf. Mach. Learn. (ICML)*, 2016, pp. 1995–2003.

[44] M. Fortunato et al., "Noisy networks for exploration," in *Proc. 6th Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–21.

[45] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[46] J. Schulman, P. Moritz, S. Levine, M. I. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," in *Proc. 4th Int. Conf. Learn. Represent. (ICLR)*, Y. Bengio and Y. LeCun, Eds., 2016, pp. 1–14.

[47] A. Schoenberg, *Theory of Harmony*. Berkeley, CA, USA: Univ. California Press, 1978.

[48] L. Bahl, P. Brown, P. De Souza, and R. Mercer, "Maximum mutual information estimation of hidden Markov model parameters for speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, vol. 1, Apr. 1986, pp. 49–52.

[49] A. Schoenberg, *The Musical Idea and the Logic, Technique and Art of Its Presentation*. Bloomington, IN, USA: Indiana Univ. Press, 2006.

[50] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2015, pp. 1889–1897.

[51] W. De Mulder, S. Bethard, and M.-F. Moens, "A survey on the application of recurrent neural networks to statistical language modeling," *Comput. Speech Lang.*, vol. 30, no. 1, pp. 61–98, 2015.

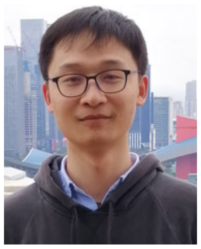
[52] B. Benward and M. Saker, *Music in Theory and Practice*. Franklin, TN, USA: Brown & Benchmark Pub, 2008.

Shulei Ji received the B.S. degree from the School of Software, Northwestern Polytechnical University, Xi'an, China, in 2019. She is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an.

Her research interests include deep music generation and music emotion recognition.

Xinyu Yang received the B.S., M.S., and Ph.D. degrees from Xi'an Jiaotong University, Xi'an, China, in 1995, 1997, and 2001, respectively.

He is currently a Professor with the School of Computer Science and Technology, Xi'an Jiaotong University. His research interests include affective computing, audio and video data processing, and automatic music generation.



Jing Luo received the B.S. degree from the School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an, China, in 2015, where he is currently pursuing the Ph.D. degree with the School of Computer Science and Technology.

His research interests include deep music generation and automatic music analysis.

Juan Li received the M.S. and Ph.D. degrees from Xi'an Jiaotong University, Xi'an, China, in 2008 and 2019, respectively.

She is currently a Professor with the Center of Music Education, Xi'an Jiaotong University. Her research interests include automatic music generation and folk music analysis.