A Study of Statistical and Music-Theoretical Melody Prediction

Huiran Yu

CMU-CS-22-153
December 2022

Computer Science Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Roger B. Dannenberg, Chair
Daniel Sleator

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science

Keywords: Melody Prediction, Statistical Models, Music theory

For Linda

Abstract

Melody prediction is an essential research focus in computer music, aiming to predict melody terms given musical context. Melody prediction can help people understand how humans form melodic anticipation while listening and also contributes to the melody generation task in automatic composition. Nowadays, most studies only focus on developing new methods to model musical sequences. However, constructing effective techniques to measure model behavior also demands attention. In our research, we offer an information entropy metric that can be applied to standard models, then further combine music theory with models to see if we can get better outcomes.

We first established a metric to measure the capability of baseline models. Each model generates a probability distribution over terms in the sequence, and we calculate the average entropy throughout the melody. Stronger models are likely to generate lower entropy, which means music is more predictable under these models. We found models trained on the whole dataset and those trained within the particular song show drastic differences. Surprisingly, training on a large dataset results in lower performance.

After setting up the baseline, we designed another model recognizing periodic occurrences of notes and patterns, incorporating music characteristics of fixed phrase length and periodic repetition of cycle position. This simple model makes satisfying predictions, and with two ensemble strategies: one considering the entropy value and confidence of each model; another one conditioned the statistical model with cycle position, we combined the new model with the statistical model reducing the prediction error from 9.9% to 6.5%.

Acknowledgments

I would like to thank my advisor Roger Dannenberg for guiding me into the world of Computer Music. All the discussions we have and the ideas we share have all been great treasures for me. And I would like to thank Daniel Sleator for his valuable advice and participation as a committee member.

I would also like to thank Shuqi Dai and Rohan Sharma for our pleasant collaboration and their contributions to the topic.

Finally, I would like to thank my family and friends for their unconditional support.

Contents

List of Figures

Chapter 1

Introduction

Melody prediction is an essential research focus in computer music, aiming to predict melody terms given musical context. Unlike music generation, which produces new music pieces, melody prediction focuses more on analyzing the expectation and surprises of existing music pieces. Previous work has shown that sequence prediction models have fundamental connections with human anticipation of music, which contributes to the understanding of human cognition.

As reported in [15] that the statistical prediction models are able to capture the information contained in melodies. Specifically, the variable-order Markov model characterizes a combination of n-gram statistics within the song by detecting different lengths of repetitions. In spite of evaluating the performance of the model with accuracy metrics calculated based on the ground truth, it was also proposed in [15] that measuring the entropy of the probability distribution given by the model can better describe the certainty of the model as well as the predictability of the test music sequence.

We should also pay attention to the training data to build a satisfying prediction model. Because of the prevalence of internal structure and repetitions within each song [6], it is revealed by our study that models trained with the same specific song of the test phrase perform much better than the models trained on the general dataset. This finding indicates that individual songs are not sampled from the overall distribution of the dataset, and the connections and references within each song are important for melody prediction. Sometimes, the overall dataset distribution has an even negative impact on the prediction result. These are all difficulties for models based on the assumption that specific behaviors like

melodies should be statistically similar to all melodies in general. In particular, neural network models are problematic because (1) they hardly consider repetitions and structures explicitly; (2) they require far more than the data within one song for training; and (3) fine-tuning to a particular melody still leaves much information from the general dataset in the model.

Given that the repetition structure in music has already been demonstrated using only statistical models without music rules and theories involved, in this work, we asked whether incorporating music features into the prediction model would further improve the model's performance. We designed a model recognizing the periodic occurrence of notes and patterns, incorporating music characteristics of fixed phrase length and periodic repetition. Then, we designed two combination methods to aggregate the merits of the new model and the statistical model and successfully reduced the error rate of prediction.

The main contributions of this work are:

- Establishing baseline performances of Markov models and variable-order Markov models on melody prediction;
- Discovering the differences between song-specific information and dataset information;
- Designing a bar-cycle repetition recognizer based on music characteristics;
- Demonstrating two ensemble strategies to combine the statistical model and the bar-cycle model, resulting in improvements over the best-known statistics-only approach.

We will introduce the baseline models in the next section, the new model in Section 4, the model performances and the ensemble strategy in Section 5, and we present conclusions in Section 6.

Chapter 2

Related Work

Expectation plays an important role in the human cognition process of the surroundings. The music cognition procedure of humans can be modeled as learning and conditional probability generation, building a bridge between information theory and music structure [13]. Music expectations psychologically influence music production [4, 18], perception [5, 11] and emotion [17]; besides melody, other music features have been tested to examine their relationship with expectation: rhythm [9, 10, 12], harmony [3, 17, 19], and so on.

Inspired by the idea from Meyer [13], Markus Pearce has found evidence for variable-order Markov process psychologically and musicologically by following how people make expectations on music [15]. He also started using entropy from the probability distribution to measure the ability of the model in the melody prediction task.

There are other machine learning methods used in music sequence prediction. Expectation Networks [21], rather than assuming a fixed prefix of length N, consider that each element of the prefix exerts an influence on expectation that decays with distance. Deep learning methods have experienced a rapid development in recent years, incorporating Variational Auto-Encoders (VAEs) [16, 23], Generative Adversarial Networks (GANs) [24] and sequential models such as LSTMs and Transformers [8, 14, 20]. However, they fail to capture long-term dependency and self-references within a song, and it is also hard for them to adjust to the statistics of a specific song [6].

Chapter 3

Baseline Models: Variable-Order Markov Chain

In this chapter, we are going to introduce the baseline model: variable-order Markov chain, which is a statistical model that combines different orders of Markov process and captures variable lengths of repetitions in the sequence. Then, we discuss the experiment settings of training data partition, confidence mechanism, and evaluation metrics.

3.1 Model Definitions

3.1.1 D-order Markov Model

Define Σ as a finite alphabet. Given a training sequence $q_1^n = q_1 q_2 \cdots q_n$, $q_i \in \Sigma$, we would like to learn the probability distribution $\hat{P}(s_n|s_1^{n-1})$ for all $s_n \in \Sigma$. Here, s_1^{n-1} represents the prediction context, which is a string prefix.

Suppose the probability distribution of the current term is only dependent on D previous observations. Then,

$$\hat{P}(s_n|s_1^{n-1}) = \hat{P}(s_n|s_{n-D}^{n-1}) \tag{3.1}$$

We estimate this distribution with the following formula:

$$\hat{P}(s_n|s_{n-D}^{n-1}) = \frac{N(s_n|s_{n-D}^{n-1})}{\sum_{\sigma\in\Sigma} N(\sigma|s_{n-D}^{n-1})} \tag{3.2}$$

where $N(\sigma|s_{n-D}^{n-1})$ is the number of times σ appears after the context s_{n-D}^{n-1}. When $N(s_n|s_{n-D}^{n-1}) = 0$, the probability will also be zero, resulting in infinity when calculating cross-entropy. Therefore, we add an initial count ϵ at each entry, and the estimation formula turns into:

$$\hat{P}(s_n|s_{n-D}^{n-1}) = \frac{N(s_n|s_{n-D}^{n-1}) + \epsilon}{\sum_{\sigma\in\Sigma}(N(\sigma|s_{n-D}^{n-1}) + \epsilon)} \tag{3.3}$$

In practice, the initial count ϵ is a hyper-parameter which needs to be carefully chosen. And when D becomes larger, this model suffers from the problem of data sparsity, and cannot fully use the subsequence repetitions, which are shorter than D, in the training data for prediction.

3.1.2 Variable-Order Markov Model

To solve the problems of data sparsity and selection of initial count in the fixed-order Markov model, we would like to merge Markov models of different orders into one prediction model. This merged model is more useful because it falls back to a lower order model when the item is not found in a higher order model.

In other words, where high-order Markov chains suffer from cases where there are no transitions from s_{n-D}^{n-1} to s_n in the training data, rather than "guessing" some small probability by adding ϵ to the count, we can fall back or "escape" to a lower order Markov chain and use it to make a more principled estimate of probabilities. When to consider lower order models and with what weight have been explored in the literature [1], but there is no optimal method.

The Prediction by Partial Match (PPM) [1] algorithm we have adopted chooses to escape when there is no transition to s_n in the training data. Thus, the lower-order model is only used to predict probabilities for the subset of the alphabet that does not appear in the training data for the higher order model. This information is important because it allows us to construct the lower-order model to predict only a subset of the whole alphabet, excluding those symbols predicted by the higher-order model. This is called the *exclusion mechanism*.

It should also be noted that the variable-order Markov Model is recursive in that after escaping to the next lower-order model, if training data is still not found to predict a transition probability, we escape again to the next lower-order model, etc., until a zero-order model is reached.

The following equations describe the PPM implementation of the variable-order Markov model. First, we show the model for the case where some zero counts exist in the training data in higher order, requiring an escape mechanism. Then we consider the special case where training data provides non-zero counts for the entire alphabet. Finally, we modify the equations to include the exclusion mechanism.

Escape Mechanism

The general formal expression for all versions of the PPM algorithm is:

$$\hat{P}(s_n|s_{n-D}^{n-1}) = \begin{cases} \hat{P}(s_n|s_{n-D}^{n-1}), & s_{n-D}^{n} \in \text{ training set} \\ \hat{P}(s_n|s_{n-D+1}^{n-1})\hat{P}(escape|s_{n-D}^{n-1}), & \text{otherwise} \end{cases} \tag{3.4}$$

We used the PPM-C variant, in which the two factors on the right of equation 3.4 are defined by the following formulas:

$$\hat{P}(\sigma|s) = \frac{N(\sigma|s)}{\sum_{\sigma' \in \Sigma(s)} N(\sigma'|s) + |\Sigma(s)|} \tag{3.5}$$

$$\hat{P}(escape|s) = \frac{|\Sigma(s)|}{\sum_{\sigma' \in \Sigma(s)} N(\sigma'|s) + |\Sigma(s)|} \tag{3.6}$$

Here, $N(\sigma|s)$ is the number of symbol σ appearing after the context s; $\Sigma(s)$ is the set of symbols that appear after the context s.

When $\Sigma(s) = \Sigma$, i.e., all possible symbols occur at least once in the training data, there will be no need to escape, and calculating the escape probability will cause $\sum_{\sigma \in \Sigma} \hat{P}(\sigma|s) \neq 1$. This mechanism is based on the assumption in general sequences that there will always be terms that are not included in the predefined alphabet Σ, but we do not need this assumption in melody prediction because all the notes are already known at the beginning. The model will never fall to a lower order in this case. Therefore, we must use a different estimation formula:

$$\hat{P}(\sigma|s) = \frac{N(\sigma|s)}{\sum_{\sigma' \in \Sigma(s)} N(\sigma'|s))}, \text{ when } \Sigma(s) = \Sigma. \tag{3.7}$$

In contrast, when σ does not appear in the entire training set, we need to assign a probability for it to keep the summation of the probabilities to one when the

recursion reaches the zero-order model:

$$\hat{P}(\sigma|\epsilon, \sigma \notin \Sigma(\epsilon)) = \frac{1}{|\Sigma - \Sigma(\epsilon)| * |\Sigma(\epsilon)|} \tag{3.8}$$

Exclusion Mechanism

When we escape to the suffix of the context s, it is no longer necessary to consider the symbols that have already appeared after the s as part of the alphabet, because we have already known that the target symbol σ will never be part of these symbols, and they can be excluded from the probability calculation. With the exclusion mechanism, if we mark the set of the excluded symbols as e, the formula (3.4)-(3.6) will turn into:

$$\hat{P}(s_n|s_{n-D}^{n-1}, e) = \begin{cases} \hat{P}(s_n|s_{n-D}^{n-1}, e), & s_{n-D}^{n} \in \text{ training set} \\ \hat{P}(s_n|s_{n-D+1}^{n-1}, e \cup \Sigma_{n-D}^{n-1}))\hat{P}(escape|s_{n-D}^{n-1}, e), & \text{otherwise} \end{cases} \tag{3.9}$$

$$\hat{P}(\sigma|s, e) = \frac{N(\sigma|s)}{\sum_{\sigma' \in \Sigma(s)/e} N(\sigma'|s) + |\Sigma(s)|)} \tag{3.10}$$

$$\hat{P}(escape|s, e) = \frac{|\Sigma(s)|}{\sum_{\sigma' \in \Sigma(s)/e} N(\sigma'|s) + |\Sigma(s)|} \tag{3.11}$$

We initialize the algorithm with $e = \emptyset$. This probability will be more accurate for it makes decision on a smaller alphabet.

The PPM algorithm is implemented with a trie as shown in Figure 3.1. Each path from the root to bottom represents a subsequence $s\sigma$ in the training data; $\Sigma(s)$ is the set of children of the last node in sequence s. During matching, we search the context from the top of the tree and when we escape, we eliminate the first term in the context sequence s and go back to the top of the tree to redo the search.

3.2 Foreground and Background

Here we define two terms: *foreground* and *background*. The foreground information is the contents within the same specific song as the predicted sequence, and the background is the set of other songs within the dataset. In practice, we randomly shuffle the dataset and separate it into a training set and a testing set for convenience. The training set is used to train the background model and every

song in the test set is trained as the foreground when testing a sequence within the song.

When predicting a sequence, we combine the probability outcome from the foreground and background models. The baseline combination is a linear combination with a mixing ratio α:

$$P_{final}(\sigma|s) = (1-\alpha)P_{foreground}(\sigma|s) + \alpha P_{background}(\sigma|s) \tag{3.12}$$

3.3 The Confidence of the Prediction

To make full use of the prediction model, we assume that predictions made by more training instances will have better reliability and introduce a confidence parameter C computed from the number of instances used to calculate the probability distribution:

$$C(P(\sigma|s)) = 1 - \frac{1}{\sum_{\sigma' \in \Sigma_s} N(\sigma'|s) + 1}, \sigma \in \Sigma(s) \tag{3.13}$$

When the number of instances equals to zero, the confidence will be zero too. As the number increases, the confidence C will approach one. To make a better balance between the two models, we only calculate the confidence of the foreground model since the number of training instances in the background will be significantly larger than the foreground, and the confidence will be so close to one that exact calculation makes little difference. The merging formula with the confidence parameter is:

$$\begin{aligned} P_{final}(\sigma|s) = {} & C(P_{foreground}(\sigma|s))(1-\alpha)P_{foreground}(\sigma|s) \\ & +(1 - C(P_{foreground}(\sigma|s))(1-\alpha))P_{background}(\sigma|s) \end{aligned} \tag{3.14}$$

3.4 Evaluation Metrics

To better inspect the probability distribution p over all the possible notes given by the model, besides prediction accuracy, we also calculated the entropy $H(p)$ and the cross-entropy $H(q,p)$ between p and the ground truth q. The entropy:

$$H(p) = -\sum_{\sigma \in \Sigma} p_\sigma \log(p_\sigma) \tag{3.15}$$

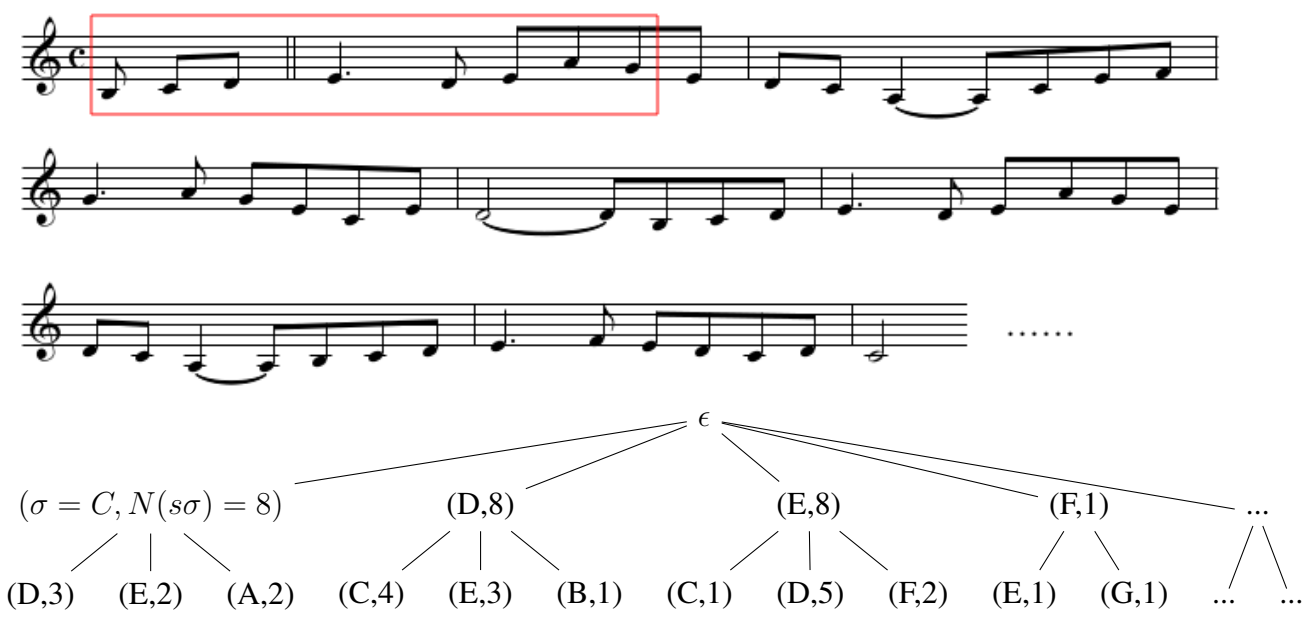

Figure 3.1: Tree constructed by PPM algorithm to predict the notes in the red box with other part of the melody sequence. The order of the variable-order Markov is one.

is independent of the ground truth σ_{gt}, which means it solely describes how certain the trained model is predicting a note after a context. The cross-entropy:

$$H(q, p) = -\sum_{\sigma \in \Sigma} q_\sigma \log(p_\sigma) = -\log(p_{\sigma_{gt}}) \tag{3.16}$$

is the unexpectedness of the model when the ground truth is revealed.

Chapter 4

The Bar-Cycle Model

4.1 Definition of the Bar-Cycle Model

One simple but significant feature of music, especially pop music, is that the content of music often repeats after some number of measures, and the repeat period is generally a power of two. The reason behind this is that music phrases, at least in popular music, tend to be organized in groups of two at different levels. So we have alternating strong and weak beats; two of these pairs make a measure; two measures form a sub-phrase; two sub-phrases form a 4-measure phrase; two of these might form a call and response, etc. Also, music content is likely to repeat itself throughout the music piece, leading to repetition at delays that are powers of two. This characteristic is deeply related to the repetition structure in music.

In music form analysis, the onset time of the note in the measure is defined as "measure position". We generalized this definition into "cycle position" to identify repetitions on larger time scales. We model this bar-cycle phenomenon as a time-position conditioned first-order Markov model as shown in Figure 4.1. Suppose we have a pitch sequence $S = [(t_1, s_1), (t_2, s_2), \cdots, (t_N, s_N))]$, where t_i is the onset time of the note, and s_i is the pitch of the note. Then,

$$P(s_i|S_1^{i-1}) = P(s_i|(t_{i-1}, s_{i-1})) = P(s_i|(\hat{t}_{i-1}, s_{i-1})), \tag{4.1}$$

Where $\hat{t}_{i-1} = t_{i-1} \mod \text{len}(cycle)$, the cycle position of the note at $i-1$. Specially, $P(s_0) = P(s_0|\hat{t}_0, \epsilon)$. In our experiments, we tested the model with cycle lengths of one measure, two measures and four measures. The onset times

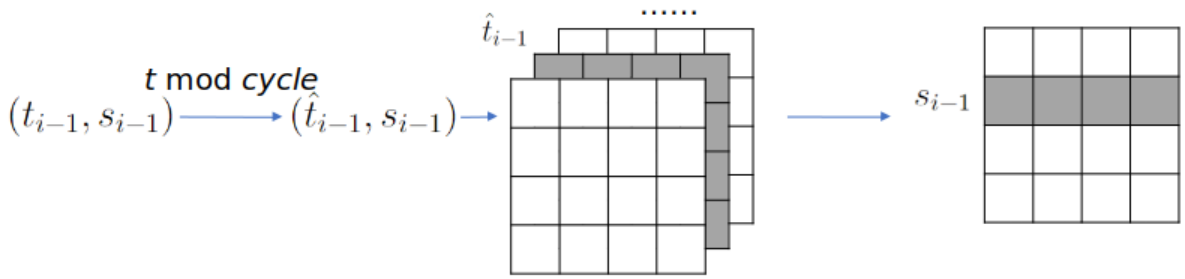

Figure 4.1: The bar-cycle model takes the onset time t_{i-1} and the pitch s_{i-1} of the context as the input and calculates its cycle position $\hat{t}_{i-1}$, then makes a prediction with the corresponding transition matrix of the cycle position.

of the notes are measured in 16-th notes.

4.2 Bar-Cycle Model and Variable-Order Markov Model Combination

We can see from previous descriptions that the bar-cycle model and the variable-order Markov model focus on different aspects of melodies. The variable-order Markov model finds different lengths of motif repetition, and the bar-cycle model focuses on repetitions at a fixed time offset. Therefore, if we can find a way to combine the prediction result of the two models, we might achieve better prediction performance.

Here we introduce an ensemble method based on the entropy and confidence properties of the predicted probability distribution. Suppose the predicted result of the variable-order Markov model P_v has the entropy of H_v with confidence C_v, and each entry is predicted under order L. Suppose the result of the bar-cycle model P_b, has the entropy of H_b with confidence C_b. Then we are going to decide which model we should believe in at each place of the test sequence. This is a typical binary classification problem, and can be solved with a Support Vector Machine [7] model.

If we denote the classification function described by SVM as $f(H_v, C_v, L, H_b, C_b)$, $f = 1$ when choosing the variable-order Markov model, $f = 0$ when choosing the bar-cycle model, then the merged probability P will be:

$$P = f(H_v, C_v, L, H_b, C_b)P_v + (1 - f(H_v, C_v, L, H_b, C_b))P_b \qquad (4.2)$$

In practice, we used the training set as background and the validation set as foreground to train the SVM classifier, and test with the test set as the foreground and the training set as the background.

4.3 Positional Variable-Order Markov Model

The previous section described an approach that merges the prediction result of two separate models. Now, in this section, we would like to take another approach where we condition the variable-order Markov model with the cycle positions of notes.

Denote $\hat{T}_c$ as the set of possible cycle positions under the cycle length of c. If we simply construct a variable-order Markov model on the alphabet of $\hat{T}_c \times \Sigma$, it will suffer from exponential state explosion as we go into higher orders. Because of this, we made an assumption that only the cycle position of the last note rather than every note of its context can affect the probability distribution of a note.

Suppose we have a pitch sequence $S = [(t_1, s_1), (t_2, s_2), \cdots, (t_N, s_N)]$, where t_i is the onset time of the note, and s_i is the pitch of the note. If the variable-order Markov model has the maximum order of D, and the cycle length is c, then we assume

$$\begin{aligned} P(s_i|S_1^{i-1}) &= P(s_i|t_{i-1}, s_1, s_2, \cdots, s_{i-1}) \\ &= P(s_i|\hat{t}_{i-1}, s_{i-D}, s_{i-D+1}, \cdots, s_{i-1}) \end{aligned} \tag{4.3}$$

Where $\hat{t}_{i-1} = t_{i-1} \mod c$ is the cycle position of the note at $i-1$. Equation 3.10 and 3.11 turn into:

$$\hat{P}(\sigma|s, \hat{t}, e) = \frac{N(\sigma|s, \hat{t})}{\sum_{\sigma' \in \Sigma(s,\hat{t})/e} N(\sigma'|s, \hat{t}) + |\Sigma(s, \hat{t})|)} \tag{4.4}$$

$$\hat{P}(escape|s, \hat{t}, e) = \frac{|\Sigma(s, \hat{t})|}{\sum_{\sigma' \in \Sigma(s)/e} N(\sigma'|s, \hat{t}) + |\Sigma(s, \hat{t})|} \tag{4.5}$$

These two equations can only be used when $\hat{t} \in \hat{T}(s_{i-1})$, where $\hat{T}(s_{i-1})$ is the set of onset times where s_{i-1} appears in the training data. If $\hat{t} \notin \hat{T}(s_{i-1})$, the model will fall back to the original variable-order Markov model.

Chapter 5

Experiments

5.1 Dataset

We used two datasets: POP909 and PDSA in our experiments.

POP909[22] is a Chinese pop song dataset that contains 879 songs in total after eliminating triple-time songs. The songs are labeled with melody, beat, chord and tonality, and they are segmented into sections and phrases. The dataset is split into training set, validation set and test set of size 529, 175, 175 respectively.

The Public Domain Song Anthology (PDSA)[2] is a lead sheet dataset containing 258 public domain pop songs, folk songs and general classical pieces. The dataset is split into training set, validation set and test set of size 156, 51, 51 respectively.

5.2 Experimental Settings

Instead of predicting the target sequences autoregressively with only prefixes training the foreground model, we included both prefix and suffix sequences as the training data. This is based on the consideration that in practice, people like to hear music multiple times, which means they will already have the impression of the whole picture of the song before they expect the next note to come. From another point of view, different from composing a music piece from scratch, structure and repetition analysis requires the information of the whole song to see the internal connections. We also consider that while there is almost

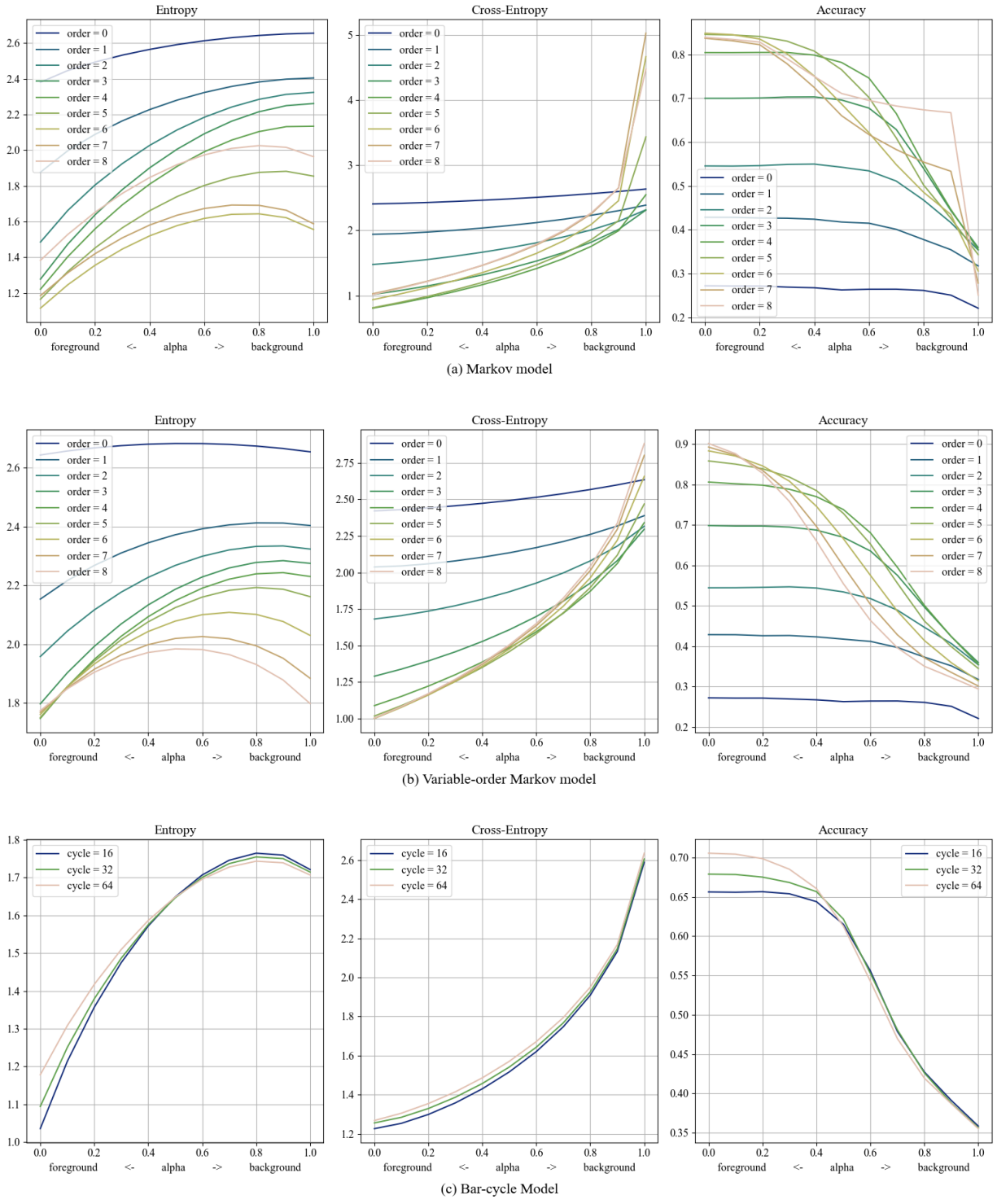

Figure 5.1: Entropy, cross-entropy and accuracy metric of Markov model, variable-order Markov model, and bar-cycle model on the POP909 dataset.

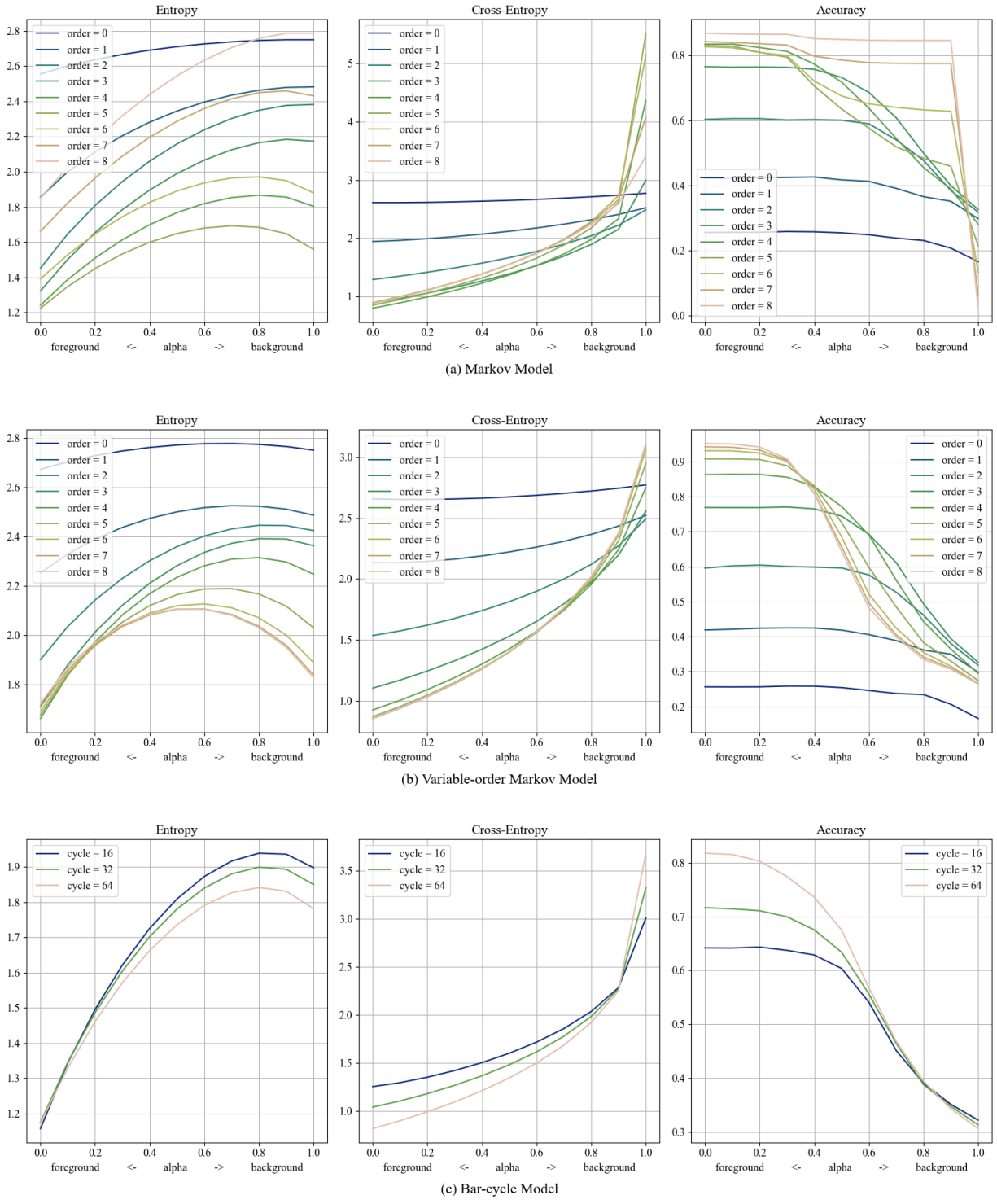

Figure 5.2: Entropy, cross-entropy and accuracy metric of Markov model, variable-order Markov model, and bar-cycle model on PDSA dataset.

no information available to predict the first few notes from prefix data, the same notes could be repeated later in the context of a substantial prefix. From a music information standpoint, should the entropy of these notes change drastically from their first occurrence to their second? On the other hand, our approach is somewhat arbitrary and we do not believe it is intrinsically "right" or "wrong."

The pitch sequences were separated into 8-note chunks to reduce the training and evaluation time to 1/8 relative to training a foreground model for each single note. At the same time, 8 is small enough that the model can usually make use of other notes in the same phrase for prediction. When training the foreground model, the test chunk will be held out and the model is trained with every other chunk in the song. The initial count of the Markov model is set to 0.0005. The order of the Markov model and the variable-order Markov model is set to 8. We used the mixing ratio α from 0 to 1, with the step of 0.1, and we used the confidence mixing strategy. All the notes in the datasets are transposed to C major and are described as scale degree, which means we have seven different types of note in total. The accidentals were moved to the nearest natural notes.

5.3 Prediction Result of the Single Models

We used the Markov model, variable-order Markov model and bar-cycle model to predict the two datasets respectively, and also calculated the entropy and cross-entropy of the distribution. The results are given in Figures 5.1 and 5.2. Generally, the variable-order Markov model outperformed the original Markov model and the bar-cycle model. The first-order bar-cycle model has similar performance as the 3rd-order variable-order Markov model. Another noticeable result is that the foreground models outperform the background models in all three metrics (except the entropy of the high-ordered variable-order Markov model on PDSA dataset), which means that the melody sequences are more predictable under the context of the same song rather than the whole dataset.

This finding is somewhat surprising because it contradicts the common wisdom that machine learning should improve with larger datasets. The failure of this wisdom in this case can be explained partially by considering that music is full of repetition, particularly within a single song, and this enhances the performance of the variable-order Markov model. However, other work [6] points to

Order	0	1	2	3	4	5	6	7	8
Count	38	310	687	964	1060	1137	1250	1370	48806
Percentage	0.07	0.56	1.24	1.73	1.91	2.04	2.25	2.46	87.7

Table 5.1: The order of prefix matching in variable-order Markov model on the POP909 dataset

additional factors such as limited within-song vocabulary compared to the general vocabulary distribution in the database. All things being equal, repetition itself will reduce the vocabulary, so these explanations are somewhat related.

5.3.1 Prediction Result Analysis of the Variable-Order Markov Model

The prediction accuracy of the 8th-order foreground variable-order Markov model reaches 92.8% on the POP909 dataset, the cross entropy gets to lower than one bit and the entropy gets to 1.454 bits (we compute entropy using log-base-2 to obtain results in bits), which means that the distribution of this model is highly aligned with the original data distribution, and it can eliminate the number of possible selections (perplexity) of the notes down to 2.7 out of 7. Because the variable-order Markov model is a mixture of different order Markov models, it outperformed the standard Markov models.

The order that the ground truth in the POP909 dataset hit in the foreground variable-order Markov is as in Table 5.1. We can see from the table that the order is concentrated in order 8, indicating that songs within POP909 are highly repetitive.

Next, let us take a look at the failure cases of the variable-order Markov model.

1. Suppose we are predicting the succeeding note after sequence (a) II-I-VI-III-III-II-I-VI in Figure 5.3.1. According to the trained foreground model, there are two possible succeeding notes: III, with two occurrences, labeled (b); and II, also with two occurrences, labeled (c). If we take a closer look at the two sequences, we will find that they have different onset times within the measure. Therefore, another condition based on measure position may distinguish these two situations.

2. Suppose we are predicting the succeeding note after sequence (a) III-V-II-III in Figure 5.3.2. As in 1., the sequences labeled (b) and (c) can be separated by onset time conditions. We can see from these two examples

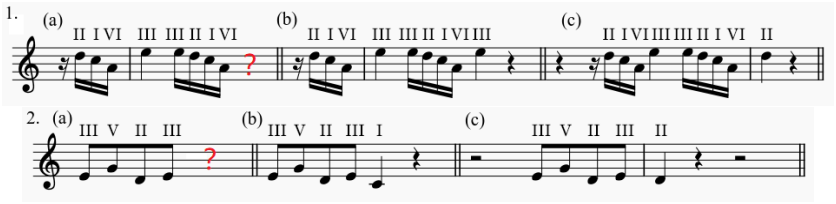

Figure 5.3: Failure cases of the variable-order Markov model. The roman numerals above the notes are the scale degree of the notes. The model is going to predict the note after sequence (a). The last note in (c) is the predicted and (b) is the ground truth.

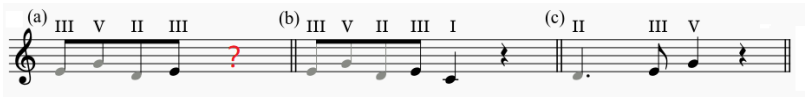

Figure 5.4: A failure case of the bar-cycle model. The roman numerals above the notes are the scale degree of the notes, and the gray notes are not visible to the model. The model is going to predict the note after sequence (a). The last note in (c) is the predicted sequence and (b) is the ground truth.

that the pure variable-order Markov model suffers when the same motives appear at different rhythm positions, no matter at which order the model matched the prefix.

5.3.2 Prediction result analysis of the bar-cycle model

If we compare the result of the bar-cycle model with the variable-order Markov model with maximum order of one, we can find that the performance improves significantly on all three metrics, which means the cycle position condition vastly improve the predictability of the melody sequence.

Now let us look at a failure case of the bar-cycle model. Suppose we are predicting the succeeding note of sequence (a) in Figure 5.4. Because the bar-cycle model is only first order, it can only see the prefix of a note III at the 6-th rhythm position in the bar, and all the gray notes are not considered. According to our foreground model, note III at the 6th rhythm position appears in two sequences in the song: (a) III-V-II-III-I, with occurrence of one; (b) II-III-V, with occurrence of two. If longer prefixes were accessible, the model would have made a correct prediction. Note that Figure 5.4 and Figure 5.3.2 are the same test sequence on which both models fail. If we can combine these two models in some way, we can enhance the prediction performance.

Model	Entropy	Cross-Entropy	ACC(%)	Entropy	Cross-Entropy	ACC(%)
		POP909			PDSA	
Bar-cycle	**1.18**	1.27	70.6	**1.15**	**0.51**	82.3
VOMM	1.53	0.79	90.1	1.65	**0.51**	95.1
SVM	1.42	0.83	93.5	1.72	0.87	95.2
Positional VOMM	1.48	**0.77**	**94.4**	1.63	0.84	**95.9**

Table 5.2: Model performances on POP909 dataset.

5.4 Combining the Bar-Cycle Model with the Variable-Order Markov Model

As we discovered that the models perform better on full foreground information, in this section we only evaluate the foreground models, with $\alpha = 0$ and confidence $C = 1$.

Table 5.2 shows the result of the baseline methods and two different combination methods on POP909 and PDSA datasets. In the POP909 dataset, the two combination methods both show improvement in prediction accuracy over the standard variable-order Markov model, and the positional variable-order Markov model reaches the highest accuracy of 94.4% with the lowest cross-entropy of 0.77, reducing 34% of the failure cases in the baseline model. In the PDSA dataset, the two combination methods both show improvement in accuracy, however cross-entropy was lower in the Bar-cycle and variable-order Markov models.

Overall, the results suggest that repetition in popular music is aligned with measures. We can make use of this property for melodic prediction, improving on the best-known position-independent prediction offered by variable-order Markov models.

Chapter 6

Conclusion

In this thesis, we evaluated statistics-based and music-characteristic-based models on a melody prediction task in terms of accuracy, entropy and cross-entropy. From the results, we discovered that:

1. Variable-order Markov models can detect different lengths of repetitions throughout the song, and has the best performance among the single models;

2. Foreground song-specific information is much more important than background dataset information in melody prediction, which is in contrast to the general perception that machine learning models performs better on larger datasets. This indicates that the materials within a song are highly repetitive and distinctive. Performance of the bar-cycle model further shows that these repetitions are ordered in a periodic manner, or at least at predictable distances such as a two measures.

3. We proposed two mixing strategies between the variable-order Markov model and the bar-cycle model, and both of them improved the accuracy of prediction. Specially, the positional variable-Markov model has reached the accuracy of 94.4%, reducing failure cases by 25% from the baseline model.

In the future, we would like combine more music-related prediction agents to the statistical models to improve prediction even further, and perhaps by doing this, we can discover more important regularities in popular music melody construction. Also, we want to use the findings in this research to inspire deep learning models to generate more convincing, structured, and human-like music.

Bibliography

[1] Ron Begleiter, Ran El-Yaniv, and Golan Yona. On prediction using variable order markov models. *Journal of Artificial Intelligence Research*, 22:385–421, 2004. 3.1.2

[2] David Berger and Chuck Israels. *The Public Domain Song Anthology*. Aperio, Charlottesville, Mar 2020. ISBN 978-1-7333543-0-1. doi: 10.32881/book2. 5.1

[3] Jamshed J Bharucha. Music cognition and perceptual facilitation: A connectionist framework. *Music perception*, 5(1):1–30, 1987. 2

[4] James C Carlsen. Some factors which influence melodic expectancy. *Psychomusicology: A Journal of Research in Music Cognition*, 1(1):12, 1981. 2

[5] Lola L Cuddy and Carole A Lunney. Expectancies generated by melodic intervals: Perceptual judgments of melodic continuity. *Perception & Psychophysics*, 57(4):451–462, 1995. 2

[6] Shuqi Dai, Huiran Yu, and Roger B Dannenberg. What is missing in deep music generation? a study of repetition and structure in popular music. *arXiv preprint arXiv:2209.00182*, 2022. 1, 2, 5.3

[7] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28, 1998. 4.2

[8] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Ian Simon, Curtis Hawthorne, Andrew M Dai, Matthew D Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer. *arXiv preprint arXiv:1809.04281*, 2018. 2

[9] Mari Riess Jones. Dynamic pattern structure in music: Recent theory and research. *Perception & psychophysics*, 41(6):621–634, 1987. 2

[10] Mari Riess Jones and Marilyn Boltz. Dynamic attending and responses to time. *Psychological review*, 96(3):459, 1989. 2

[11] Carol L Krumhansl. Effects of musical context on similarity and expectancy. *Systematische musikwissenschaft*, 3(2):211–250, 1995. 2

[12] Edward W Large and Mari Riess Jones. The dynamics of attending: How people track time-varying events. *Psychological review*, 106(1):119, 1999. 2

[13] Leonard B. Meyer. Meaning in music and information theory. *The Journal of Aesthetics and Art Criticism*, 15(4):412–424, 1957. ISSN 00218529, 15406245. URL `http://www.jstor.org/stable/427154`. 2

[14] Christine Payne. Musenet. *OpenAI, openai.com/blog/musenet*, 2019. 2

[15] Marcus T Pearce and Geraint A Wiggins. Auditory expectation: the information dynamics of music perception and cognition. *Topics in cognitive science*, 4(4):625–652, 2012. 1, 2

[16] Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. A hierarchical latent vector model for learning long-term structure in music. In *Proc. of the International conference on machine learning*, pages 4364–4373. PMLR, 2018. 2

[17] Nikolaus Steinbeis, Stefan Koelsch, and John A Sloboda. The role of harmonic expectancy violations in musical emotions: Evidence from subjective, physiological, and neural responses. *Journal of cognitive neuroscience*, 18(8):1380–1393, 2006. 2

[18] William Forde Thompson, Lola L Cuddy, and Cheryl Plaus. Expectancies generated by melodic intervals: Evaluation of principles of melodic implication in a melody-completion task. *Perception & Psychophysics*, 59(7): 1069–1076, 1997. 2

[19] Barbara Tillmann, Jamshed J Bharucha, and Emmanuel Bigand. Implicit learning of tonality: a self-organizing approach. *Psychological review*, 107 (4):885, 2000. 2

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 2

[21] Niels J Verosky. Corpus-based learning of tonal expectations with expectation networks. *Journal of New Music Research*, 48(2):145–158, 2019. 2

[22] Ziyu Wang, Ke Chen, Junyan Jiang, Yiyi Zhang, Maoran Xu, Shuqi Dai, Guxian Bin, and Gus Xia. Pop909: A pop-song dataset for music arrangement generation. In *Proc. of 21st Int. Conference on Music Information Retrieval Conf.*, 2020. 5.1

[23] Shiqi Wei and Gus Xia. Learning long-term music representations via hierarchical contextual constraints. In *Proc. of the 22nd Int. Society for Music Information Retrieval Conf.*, 2021. 2

[24] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 2