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

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. This simple model makes satisfying predictions, and with an ensemble strategy considering the entropy value and confidence of each model, we combined the new model with a statistic model reducing the prediction error from 7.5% to 6.5%, eliminating 13% of the failure cases.

Contents

1 Introduction **1**

2 Baseline Models: Variable-order Markov Chain **3**
2.1 Model Definitions 3
2.1.1 D-order Markov Model 3
2.1.2 Variable-order Markov Model 4
2.2 Foreground and Background 6
2.3 The confidence of the prediction 6

3 Bar-cycle model **9**
3.1 Definition of the bar-cycle model 9
3.2 Bar-cycle model and variable-order Markov model combination 10
3.3 Engraving bar-cycle model into the variable-order Markov model 10

4 Experiments **13**
4.1 Dataset 13
4.2 Experiment Settings 13
4.3 Prediction result of the single models 16
4.3.1 The effect of confidence strategy 18
4.3.2 Prediction result analysis of the variable-order Markov model 18
4.3.3 Prediction result analysis of the bar-cycle model 20
4.4 Combining bar-cycle model with variable-order Markov model 20

5 Conclusion **23**

Bibliography **25**

List of Figures

2.1 Tree constructed by PPM algorithm to predict the notes in the red box. The order of the variable-order Markov is one. 8

4.1 Entropy, cross-entropy and accuracy metric of Markov model, variable-order Markov model, and bar-cycle model on POP909 dataset. 14
4.2 Entropy, cross-entropy and accuracy metric of Markov model, variable-order Markov model, and bar-cycle model on PDSA dataset. 15
4.3 Variable-order Markov chain on (a) POP909 and (b) PDSA dataset with linear combination strategy between foreground and background 17
4.4 Failure cases of the variable-order Markov model. The roman numerals above the notes are the scale degree of the notes. 19
4.5 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. . . . 20

Chapter 1

Introduction

Melody prediction is an essential research focus in computer music, aiming to predict melody terms given musical context. Unlike music generation (composition), which produces new music pieces, melody prediction has looser constraints on the conditions and focuses more on the analysis of the expectation and surprises of music pieces.

First, we need evaluation metrics to choose a proper model to conduct the prediction task. Beside the accuracy metric which only describes the final outcome, in this study we selected two other metrics: entropy and cross-entropy. Entropy can measure the uncertainty of a prediction, and cross-entropy can measure the difference between the real distribution and the predicted distribution. To some extent, entropy describes how certain is the model to the prediction result and the cross-entropy describes the degree of surprise from the model when the answer is revealed. A good prediction model will have low entropy and cross-entropy while maintaining high accuracy.

To build a satisfying prediction model, we should also pay attention to the training data of the model, for it has a great impact on the performance. Because of the prevalence of internal structure and repetitions within each song, 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 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. Instead, group of statistical models are used in our study because they can be trained easily with an arbitrary number of instances and we can easily track which training data has made the model predict the probability distribution. With a proper model selected by the accuracy, entropy and cross-entropy metrics, melody prediction can help people model how humans form melodic anticipation while listening and also contributes to the melody generation task in automatic composition.

Given the fact that the repetition structure in music has already been clearly demonstrated using only statistical models without music rules and theories involved, we would like to see whether incorporating music features into the prediction model would further improve the model performance. We designed a model recognizing periodic occurrences of notes and patterns, incorporating music characteristics of fixed phrase length and periodic repetition. Finally, we combined the new model with the statistical model based on their entropy and confidence feature and successfully reduced the error rate of prediction.

The main contribution 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 statistic 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 3, the model performances and the ensemble strategy in Section 4, and we present conclusions in Section 5.

Chapter 2

Baseline Models: Variable-order Markov Chain

2.1 Model Definitions

2.1.1 D-order Markov Model

Define Σ as a finite alphabet. Given a 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.

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

$$\hat{P}(s_n|s_1^{n-1}) = \hat{P}(s_n|s_{n-D}^{n-1}) \tag{2.1}$$

We estimate this distribution with the following formula:

$$\hat{P}(s_n|s_{n-D}^{n-1}) = \frac{N(s_n|s_{n-D}^{n-1})}{\sum_{\sigma\in\Sigma} N(\sigma|s_{n-D}^{n-1})} \tag{2.2}$$

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:

$$\hat{P}(s_n|s_{n-D}^{n-1}) = \frac{N(s_n|s_{n-D}^{n-1}) + \epsilon}{\sum_{\sigma\in\Sigma}(N(\sigma|s_{n-D}^{n-1}) + \epsilon)} \tag{2.3}$$

In practice, the initial count ϵ is a hyper-parameter which need 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.

2.1.2 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 flexible towards variable lengths of repetitions in the sequence by falling back to lower order model when the item is not found in 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 [**?**], but there is no optimal method.

The Prediction by Partial Match (PPM) 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 Prediction by Partial Match (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, 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 formalized expressions are

$$\hat{P}(s_n|s_{n-D}^{n-1}) = \begin{cases} \hat{P}(s_n|s_{n-D}^{n-1}), & s_{n-D}^{n} \in \text{ training set} \\ \hat{P}(s_n|s_{n-D+1}^{n-1})\hat{P}(escape|s_{n-D}^{n-1}), & \text{otherwise} \end{cases} \tag{2.4}$$

where:

$$\hat{P}(\sigma|s) = \frac{N(\sigma|s)}{\sum_{\sigma' \in \Sigma(s)} N(\sigma'|s) + |\Sigma(s)|} \tag{2.5}$$

$$\hat{P}(escape|s) = \frac{|\Sigma(s)|}{\sum_{\sigma' \in \Sigma(s)} N(\sigma'|s) + |\Sigma(s)|} \tag{2.6}$$

Specially,

$$\hat{P}(\sigma|\epsilon, \sigma \notin \Sigma(\epsilon)) = \frac{1}{|\Sigma - \Sigma(\epsilon)| * |\Sigma(\epsilon)|} \tag{2.7}$$

Here, $N(\sigma|s)$ is the number of the symbol σ that appears after the context s; $\Sigma(s)$ is the set of symbols that appear after the context s. The formula at 2.7 is the special case when σ does not appear in the training set, we need to assign a probability for it to keep the summation of the probabilities to one.

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 this will cause $\sum_{\sigma \in \Sigma} \hat{P}(\sigma|s) \neq 1$. This 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 lower order in this case. Therefore, we must use a different estimation formula for this case:

$$\hat{P}(\sigma|s) = \frac{N(\sigma|s)}{\sum_{\sigma' \in \Sigma(s)} N(\sigma'|s))}, \text{ when } \Sigma(s) = \Sigma. \tag{2.8}$$

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 (2.4)-(2.6) will turn into:

$$\hat{P}(s_n|s_{n-D}^{n-1}, e) = \begin{cases} \hat{P}(s_n|s_{n-D}^{n-1}, e), & s_{n-D}^{n} \in \text{ training set} \\ \hat{P}(s_n|s_{n-D+1}^{n-1}, e \cup \Sigma_{n-D}^{n-1}))\hat{P}(escape|s_{n-D}^{n-1}, e), & \text{otherwise} \end{cases} \tag{2.9}$$

$$\hat{P}(\sigma|s, e) = \frac{N(\sigma|s)}{\sum_{\sigma' \in \Sigma(s)/e} N(\sigma'|s) + |\Sigma(s)|)} \tag{2.10}$$

$$\hat{P}(escape|s, e) = \frac{|\Sigma(s)|}{\sum_{\sigma' \in \Sigma(s)/e} N(\sigma'|s) + |\Sigma(s)|} \tag{2.11}$$

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

The PPM algorithm is implemented with a trie as shown in Figure 2.1. 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.

2.2 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 with in the dataset. In practice, we randomly shuffled the dataset and separated it into training set and testing set for convenience. The training set is used to train the background model and every song in the testset is trained as the foreground when testing a sequence within the song.

2.3 The confidence of the prediction

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

with a mixing ratio α:

$$P_{final}(\sigma|s) = (1-\alpha)P_{foreground}(\sigma|s) + \alpha P_{background}(\sigma|s) \tag{2.12}$$

To make full use of the prediction model, we introduce a confidence parameter C computed from the number of instances used to calculate the probability distribution:

$$C(P(\sigma|s)) = 1 - \frac{1}{\sum_{\sigma' \in \Sigma_s} N(\sigma'|s) + 1}, \sigma \in \Sigma(s) \tag{2.13}$$

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:

$$\begin{aligned} P_{final}(\sigma|s) = {} & C(P_{foreground}(\sigma|s))(1-\alpha)P_{foreground}(\sigma|s) \\ & +(1 - C(P_{foreground}(\sigma|s))(1-\alpha))P_{background}(\sigma|s) \end{aligned} \tag{2.14}$$

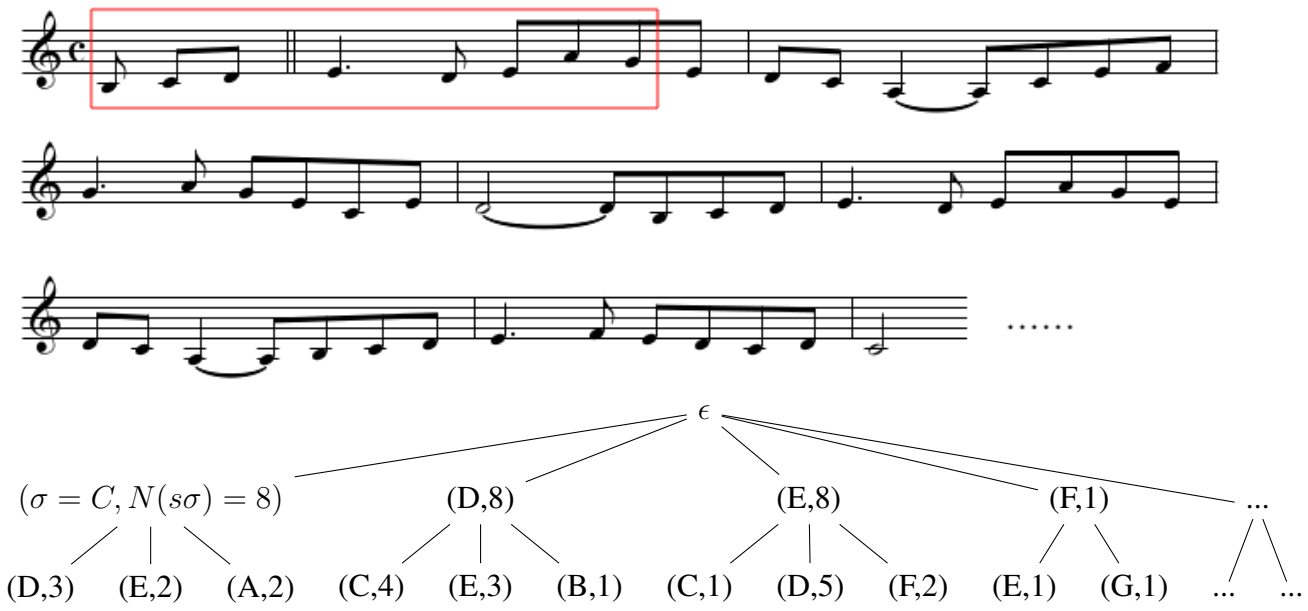


Figure 2.1: Tree constructed by PPM algorithm to predict the notes in the red box. The order of the variable-order Markov is one.

Chapter 3

Bar-cycle model

3.1 Definition of the bar-cycle model

One simple but significant feature of music, especially pop music is that the contents of music often repeat 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 form a call and response, etc. 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.

We model this bar-cycle phenomenon as a time-position conditioned first-order Markov model. 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,

$$P(s_i|S_1^{i-1}) = P(s_n|(t_{i-1}, s_{i-1})) = P(s_i|(\hat{t}_{i-1}, s_{i-1})), \tag{3.1}$$

Where $\hat{t}_{i-1} = t_{i-1} \mod \text{len}(cycle)$. Specially, $P(s_0) = P(s_0|\hat{t}_0, \epsilon)$. In our experiments, we tested the model with cycle length of one measure, two measures and four measures. The onset times of the notes are measured in 16-th notes.

(figure)

3.2 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 acquire higher 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**??** 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:

$$P = f(H_v, C_v, L, H_b, C_b)P_v + (1 - f(H_v, C_v, L, H_b, C_b))P_b \tag{3.2}$$

3.3 Engraving bar-cycle model into the variable-order Markov model

Previous section merges the prediction result of the two models, and now we would like to take one step further by engraving the bar-cycle mechanism into the variable-order Markov model.

Denote $\hat{T}_c$ as the set of possible onset times under the cycle length of c. If we simply construct a variable-order Markov model on the set of $\hat{T}_c \times \Sigma$, it will suffer from exponential state explosion as we go into higher order. Then we need to make an assumption that the probability distribution at a note is only affected by the onset time of the last note of its context.

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,

$$\begin{aligned} P(s_i|S_1^{i-1}) &= P(s_i|s_1, s_2, \cdots, s_{i-1}, t_{i-1}) \\ &= P(s_i|s_{i-D}, s_{i-D+1}, \cdots, s_{i-1}, \hat{t}_{i-1}) \end{aligned} \tag{3.3}$$

Where $\hat{t}_{i-1} = t_{i-1} \mod c$. Equation 2.10 and 2.11 turn into:

$$\hat{P}(\sigma|s, \hat{t}, e) = \frac{N(\sigma|s, \hat{t})}{\sum_{\sigma' \in \Sigma(s,\hat{t})/e} N(\sigma'|s, \hat{t}) + |\Sigma(s, \hat{t})|)} \tag{3.4}$$

$$\hat{P}(escape|s, \hat{t}, e) = \frac{|\Sigma(s, \hat{t})|}{\sum_{\sigma' \in \Sigma(s)/e} N(\sigma'|s, \hat{t}) + |\Sigma(s, \hat{t})|} \tag{3.5}$$

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 4

Experiments

4.1 Dataset

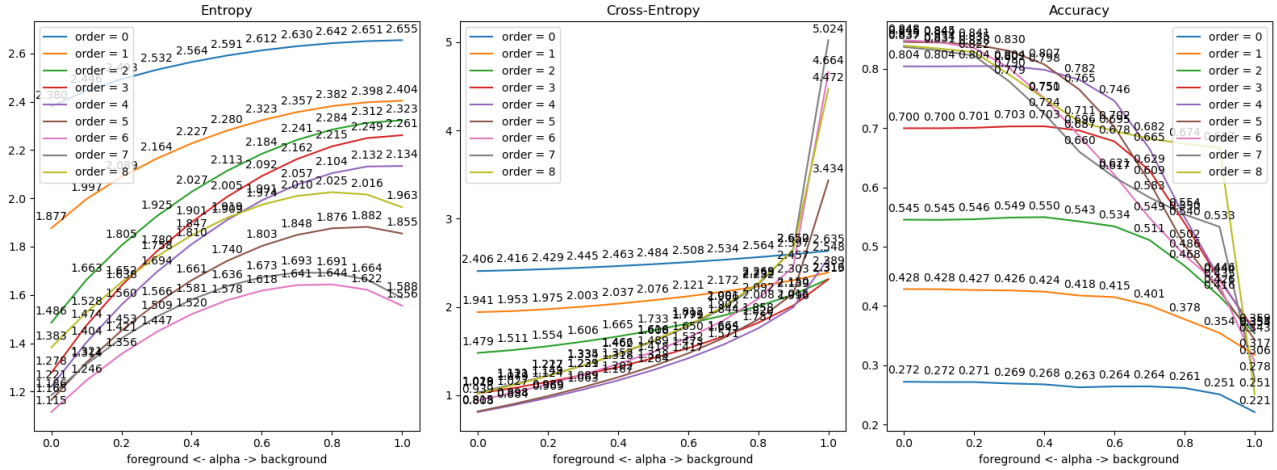
We used two datasets: POP909 and PDSA in our experiments.

POP909[2] 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.

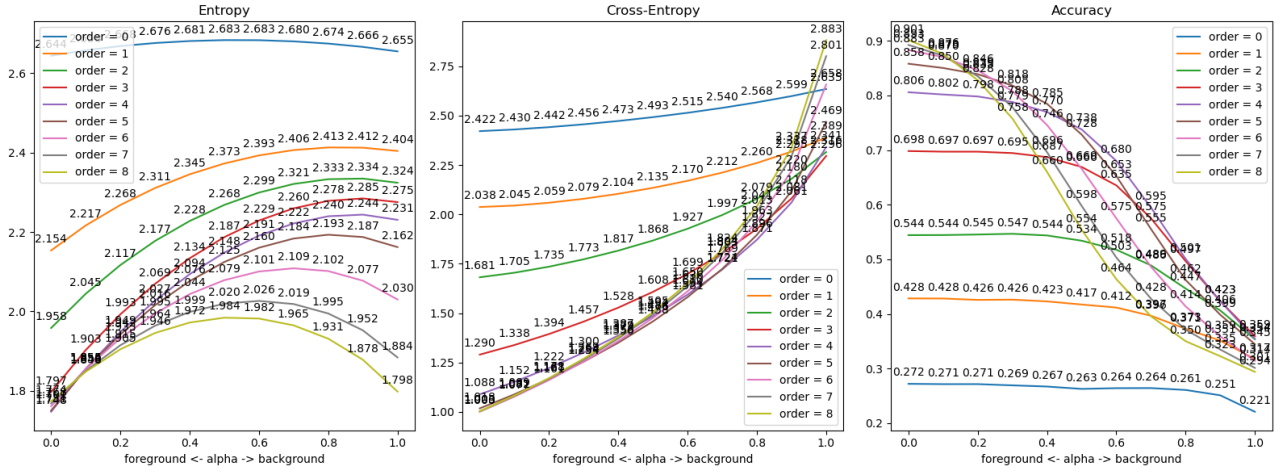
The Public Domain Song Anthology (PDSA)[1] is a lead sheet dataset containing 258 public domain pop songs, folk songs and general classical pieces. The dataset is split into training set, validation set and test set of size 156, 51, 51 respectively.

4.2 Experiment Settings

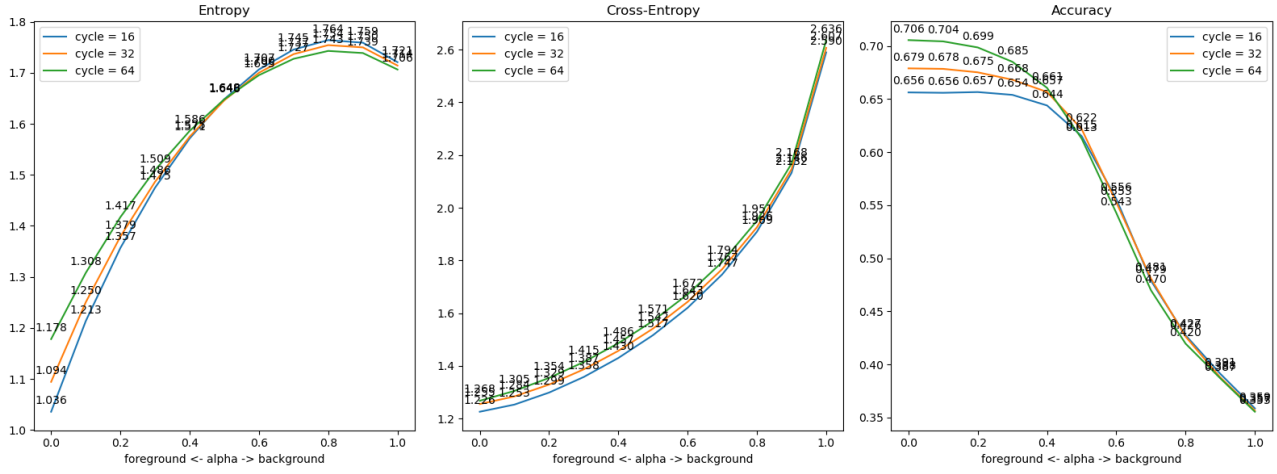
Instead of predicting the target sequences autoregressively with only prefixes training the foreground model, we included both prefix and suffix sequences as the training data. This is based on the consideration that in practice, people like to hear music multiple times, which means they will already 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



(a) Markov chain

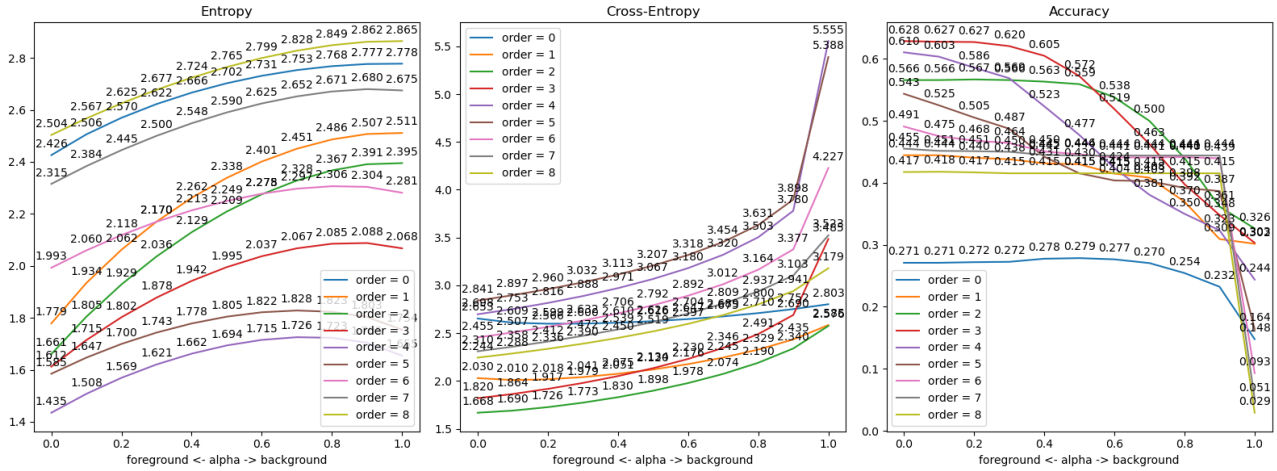


(b) Variable-order Markov model

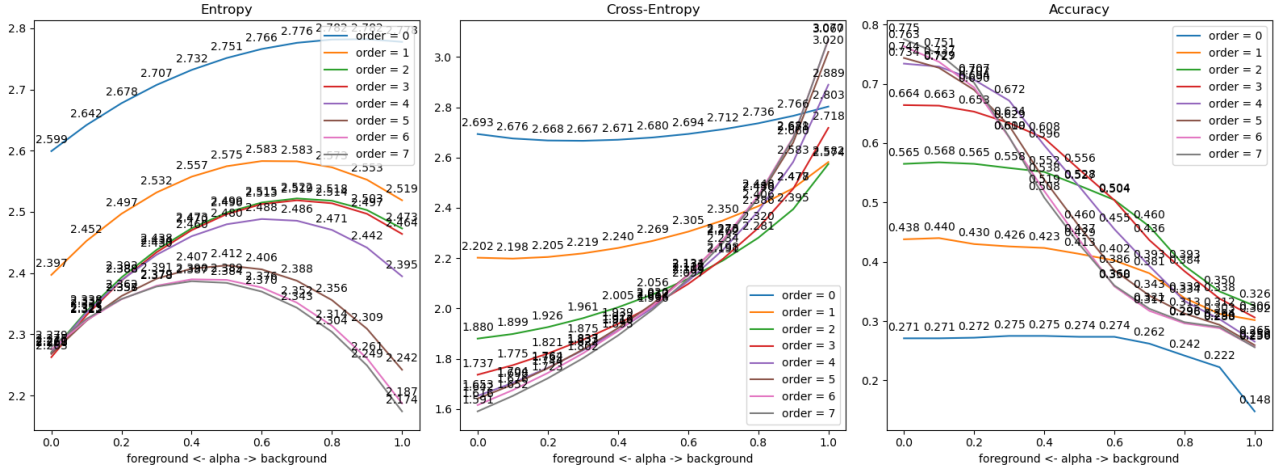


(c) Bar-cycle model

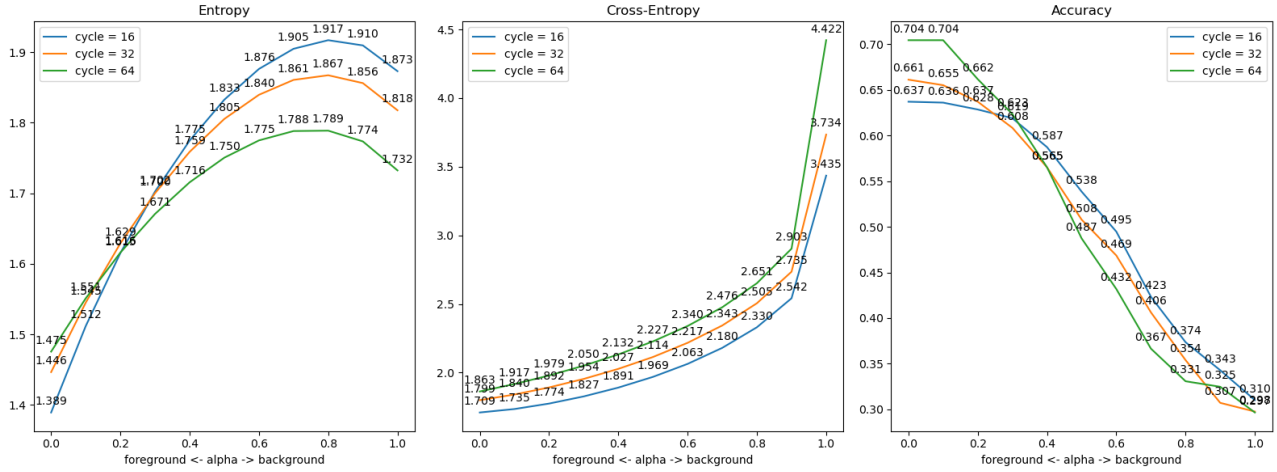
Figure 4.1: Entropy, cross-entropy and accuracy metric of Markov model, variable-order Markov model, and bar-cycle model on POP909 dataset.



(a) Markov chain



(b) Variable-order Markov model



(c) Bar-cycle model

Figure 4.2: Entropy, cross-entropy and accuracy metric of Markov model, variable-order Markov model, and bar-cycle model on PDSA dataset.

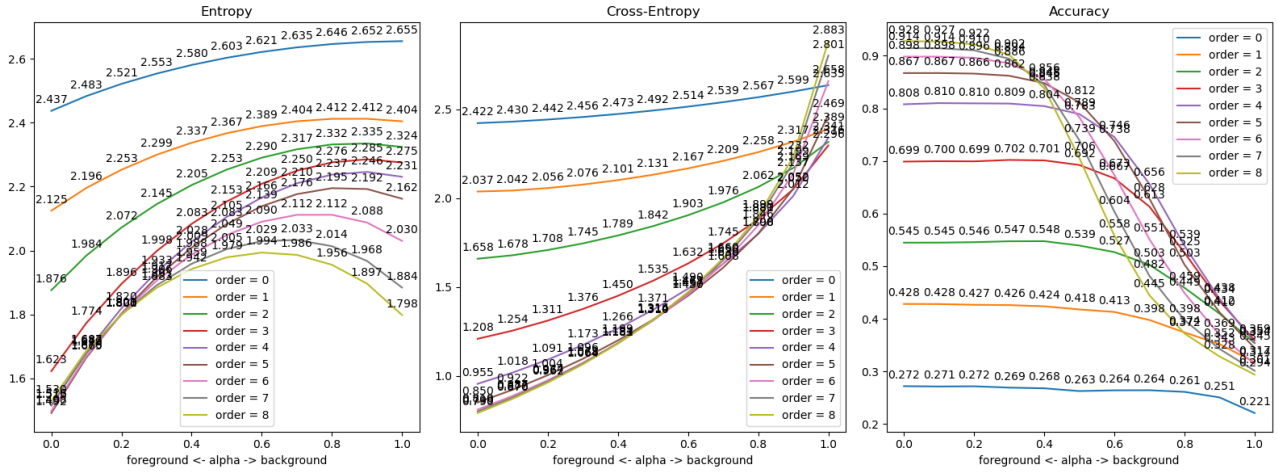
no information available to predict the first few notes from prefix data, the same notes could be repeated later in the context of a substantial prefix. From a music information standpoint, should the entropy of these notes change drastically from their first occurrence to their second? On the other 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 in each song to make full use of the notes within the same phrase while training the foreground model while reducing the training and evaluation time to 1/8 if we train a foreground model on every single note. 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.

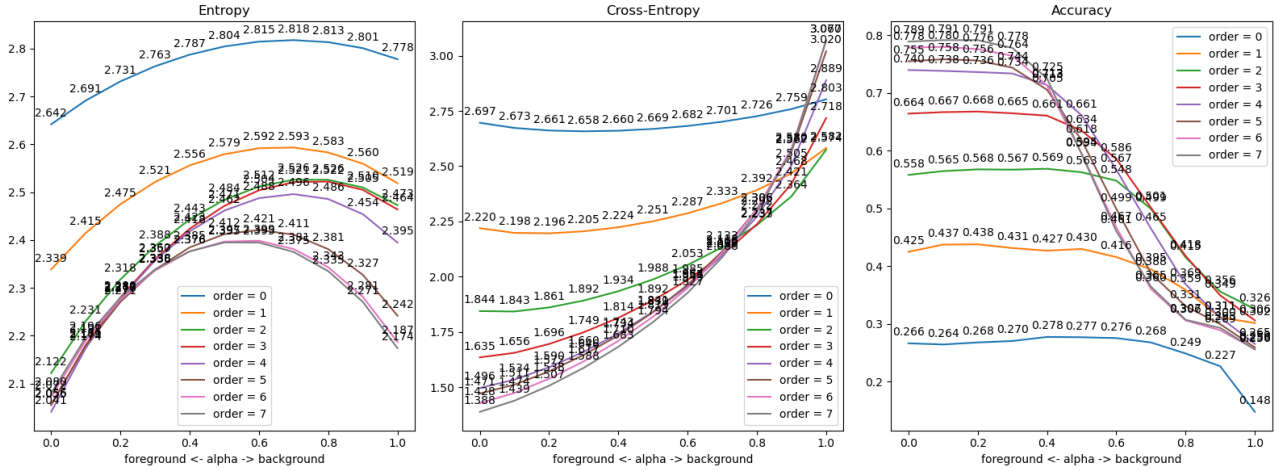
4.3 Prediction result of the single models

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 4.1 and 4.2. 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 [**? ? ?**] points to additional factors such as limited within-song vocabulary compared to the general vocabulary distribution in the database.



(a) POP909



(b) PDSA

Figure 4.3: Variable-order Markov chain on (a) POP909 and (b) PDSA dataset with linear combination strategy between foreground and background

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 4.1: The order of prefix matching in variable-order Markov model on POP909 dataset

4.3.1 The effect of confidence strategy

Figure 4.3 demonstrates the result of variable-order Markov model on the two datasets with linear foreground-background combination strategy. For almost all the mixing ratios and orders, the confidence strategy improves the accuracy with a little sacrifice on entropy and cross-entropy. However, when the maximum order of the variable-order Markov model is high enough, the confidence strategy will harm the performance of the model.

If we analyze the confidence mixing equation 2.14, we can see that because the confidence of the foreground model will always be less than one, the ratio of the foreground model will always be mapped to a point less than $(1-\alpha)$. Therefore, if we want the confidence strategy to take effect, we will need at least a portion of function 2.12 being incremental with respect to alpha. In contrast, the curve of high-ordered variable-order Markov model is monotonic decreasing on [0,1], so the confidence parameter does not take effect. This is also an evidence for the existence of long-term repetitions within a music piece because high-ordered foreground models have the best performances.

4.3.2 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 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 true term in POP909 dataset hit in the foreground variable-order Markov is as in the Table 4.1. We can see from the table that the order

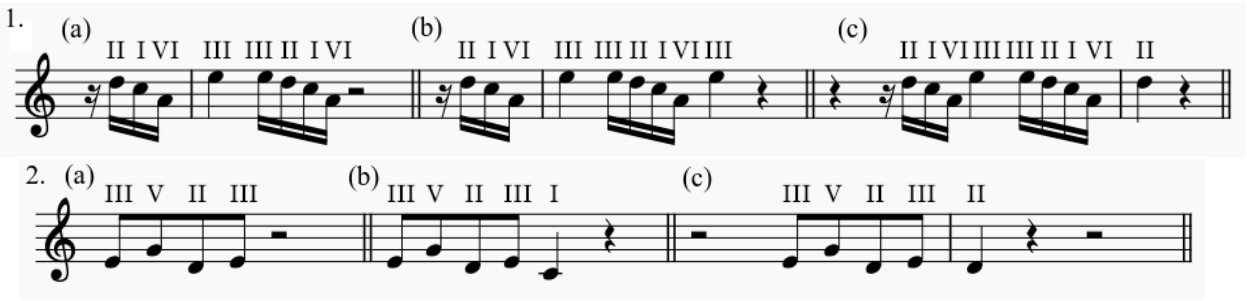


Figure 4.4: Failure cases of the variable-order Markov model. The roman numerals above the notes are the scale degree of the notes.

is concentrated in order 8, indicating that songs within POP909 are a highly repetitive.

Then, let's 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 4.4.1. According to the trained foreground model, there are two possible succeeding notes: (b) III, with the occurrence of two; (c) II, also with the occurrence of two. If we take a closer look at the two sequences, we will find that they have different onset time within the measure. Therefore, another condition based on rhythmic feature may distinguish these two situations.

2. Suppose we are predicting the succeeding note after sequence (a) III-V-II-III in Figure 4.4.2. As in 1., the sequences in (b) and (c) can be separated by onset time conditions. We can see from these two examples 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.

3. Other particular failed cases for variable-order Markov model are the beginning notes of sections, for there is no previous context and the model will only use the general distribution of the notes to make prediction. This phenomenon will continue affect the prediction accuracy until the model gets a long-enough prefix.

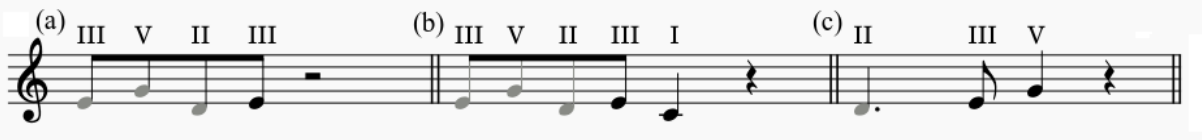


Figure 4.5: A failure case of the bar-cycle model. The roman numerals above the notes are the scale degree of the notes, and the gray notes are not visible to the model.

Model	Entropy	Cross-Entropy	Accuracy(%)
Bar-cycle	**1.18**	1.27	70.6
Variable-order Markov model	1.53	0.79	92.8
SVM combination	1.42	0.83	93.5
Cycled variable-order Markov	1.48	**0.77**	**94.4**

Table 4.2: Model performances on POP909 dataset.

4.3.3 Prediction result analysis of the bar-cycle model

If we compare the result of 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 onset time condition vastly improve the predictability of the melody sequence.

Now let's look at a failure case of the bar-cycle model. Suppose we are predicting the succeeding note of sequence (a) in Figure 4.5. Because the Bar-Cycle model is only first order, it can only see the prefix of a note III at the 6th 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 4.5 and Figure 4.4.2 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.

4.4 Combining bar-cycle model with 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 4.4 shows the result of the baseline methods and two different combination methods on POP909 dataset. The two combination methods both show improvement on the standard variable-order Markov model, and the cycled variable-order Markov model reaches the highest accuracy of 94.4% with the lowest cross-entropy of 0.77, reducing 25% of the failure cases in the baseline model.

The results also suggest that combined with information of periodic note occurrences which is related to the construction of music phrases, the melody of music is highly predictable based on song-specific content.

Chapter 5

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. The confidence mechanism, which forms a combination of foreground and background models, performs better when the mixing ratio is other than the pure foreground model, which is not the case of the best foreground models.

4. 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 bar-cycle engraving method has reached the accuracy of 94.4%, reduced 25% failure cases 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] David Berger and Chuck Israels. *The Public Domain Song Anthology*. Aperio, Charlottesville, Mar 2020. ISBN 978-1-7333543-0-1. doi: 10.32881/book2. 4.1

[2] 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. 4.1