A Study of Statistical and Music-Theoretical Melody Prediction

Huiran Yu

December 2022

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 Doctor of Philosophy.*

Keywords: Stuff, More Stuff

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

List of Figures

Chapter 1

Introduction

Melody prediction is an essential research focus in computer music, aiming to predict melody terms given musical context. Different from generation task 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 describe the final outcome, in this study we selected another two 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. Because of this observation, we did not test neural network models for (1) they hardly consider repetitions and structures explicitly; (2) the amount of data within one song is far from enough to train any existing deep learning systems; (3) fine-tuning will still get the information from the general dataset involved in the model. Instead, group of statistics models are used in our study for they can be easily trained with 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 discovered with entire 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 statistic model based on their entropy and confidence feature, successfully reduced the error rate of prediction.

The main contribution of this work are:

- Evaluated the baseline performances of Markov model and variable-order Markov model on melody prediction;
- Discovered the difference between song-specific information and dataset information;
- Designed a bar-cycle repetition recognizer based on music characteristics;
- Proposed a ensemble strategy to combine the statistic model and the bar-cycle model.

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 Dth-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 occurrence of 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.

We use Prediction by Partial Match (PPM) to implement the variable-order Markov model, which introduces escape mechanism and exclusion mechanism to unify the distributions between the models.

1. 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,

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

   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.

   When $\Sigma(s) = \Sigma$, the escape mechanism will cause $\sum_{\sigma \in \Sigma} \hat{P}(\sigma|s) \neq 1$, for the escape option is also part of the probability distribution. 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 for all the notes are already known at the beginning. The model will never fall to lower order in this case. Then, the estimation formula becomes:

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

2. 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 ϵ, the formula (2.4)-(2.6) will turn into:

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

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

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

   This probability will be more accurate for it makes decision on smaller alphabet.

The PPM algorithm is implemented with a trie as the example 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 fully use of the predicting 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 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 makes little difference. The merging formula with the confidence parameter is:

$$P_{final}(\sigma|s) = C(P_{foreground}(\sigma|s))(1-\alpha)P_{foreground}(\sigma|s) \tag{2.14}$$

$$+(1 - C(P_{foreground}(\sigma|s))(1-\alpha))P_{background}(\sigma|s) \tag{2.15}$$

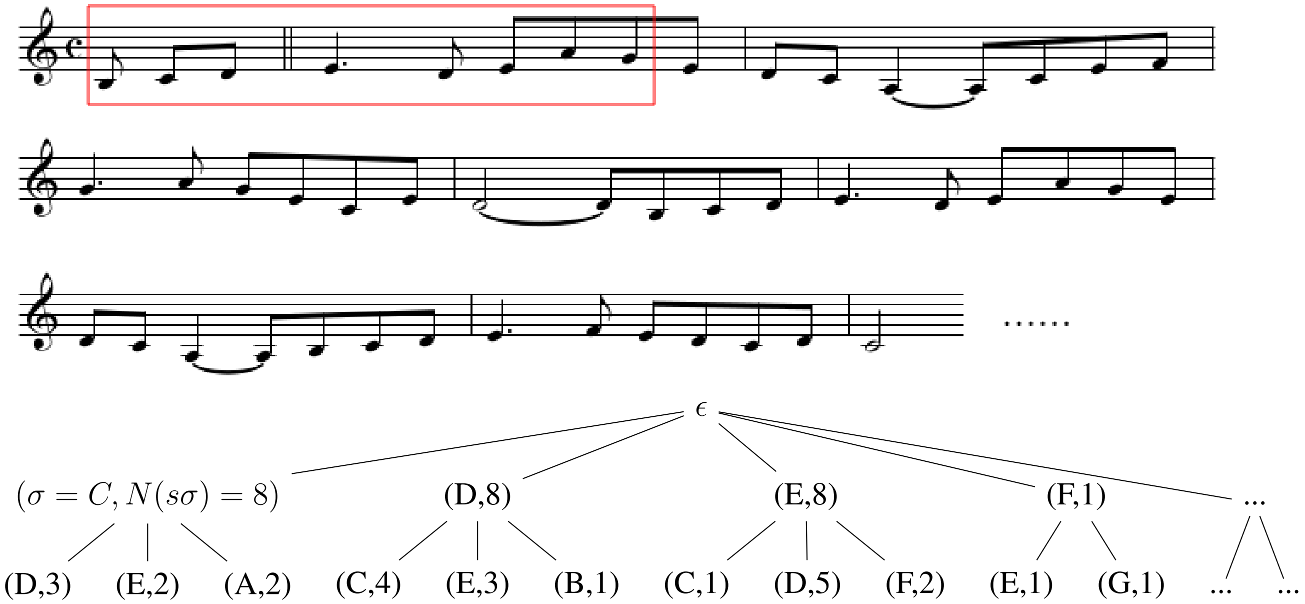

Figure 2.1: Tree constructed by PPM algorithm to predict the notes in the red box. The order of the variable-order Markov is one.

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 the power of two. The reason behind this is that the music phrases tend to have length of power of two measures, and the the contents are likely to repeat itself throughout the music piece. This characteristics deeply related to the repetition structure in music.

We model this bar-cycle phenomenon into 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, 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 measure and four measures. The onset time of the notes are measured in 16-th note.

(figure)

Chapter 4

Experiments

4.1 Dataset

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

POP909[**?**] is a Chinese pop song dataset which contains 879 songs in total. 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)[**?**] is a lead sheet dataset contains 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 for 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.

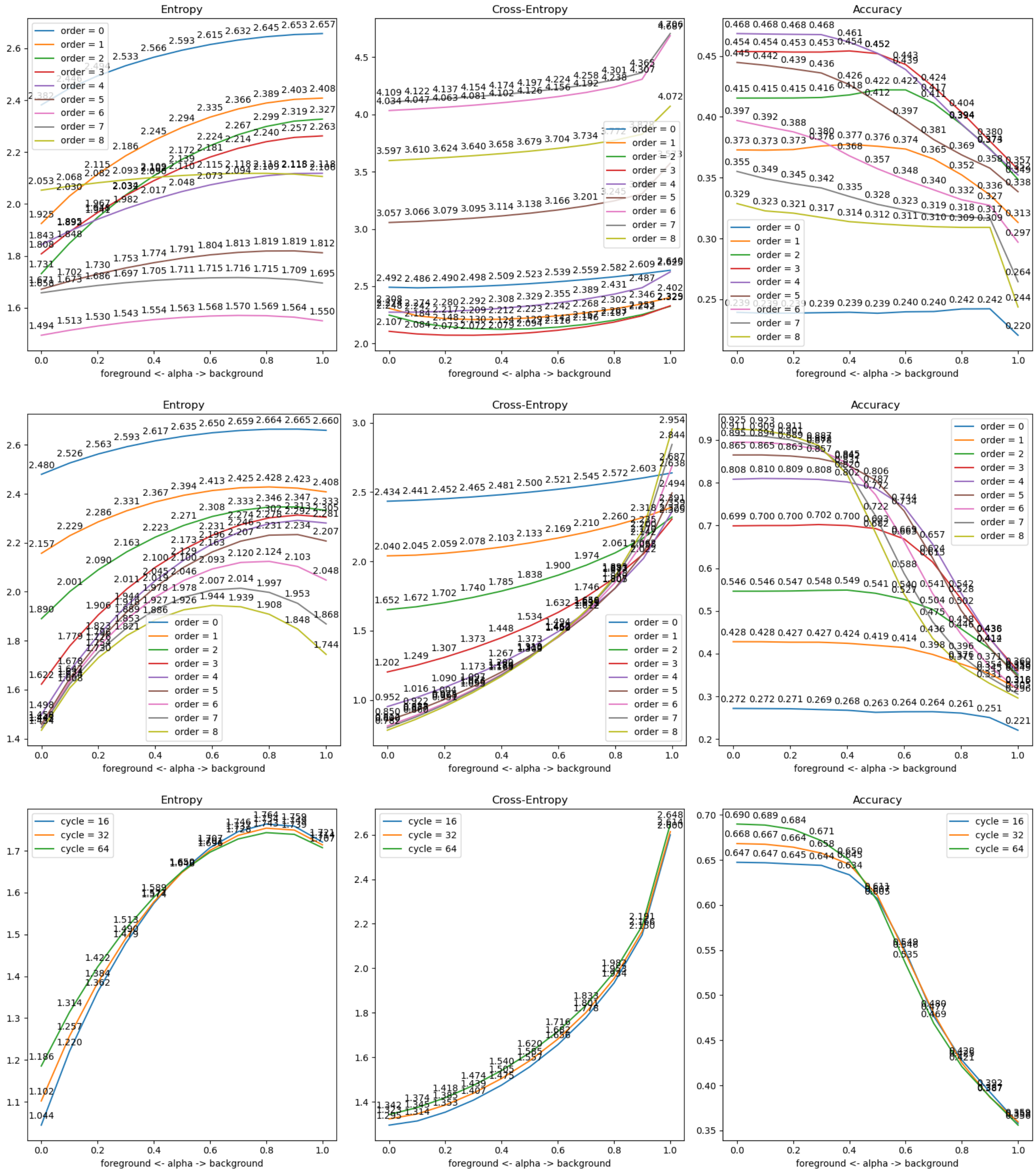

Figure 4.1: Entropy, cross-entropy and accuracy metric of Markov model, variable-order Markov model, and bar-cycle model on POP909 dataset.

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. The initial count of the Markov model is set to 0.005. The order of the Markov

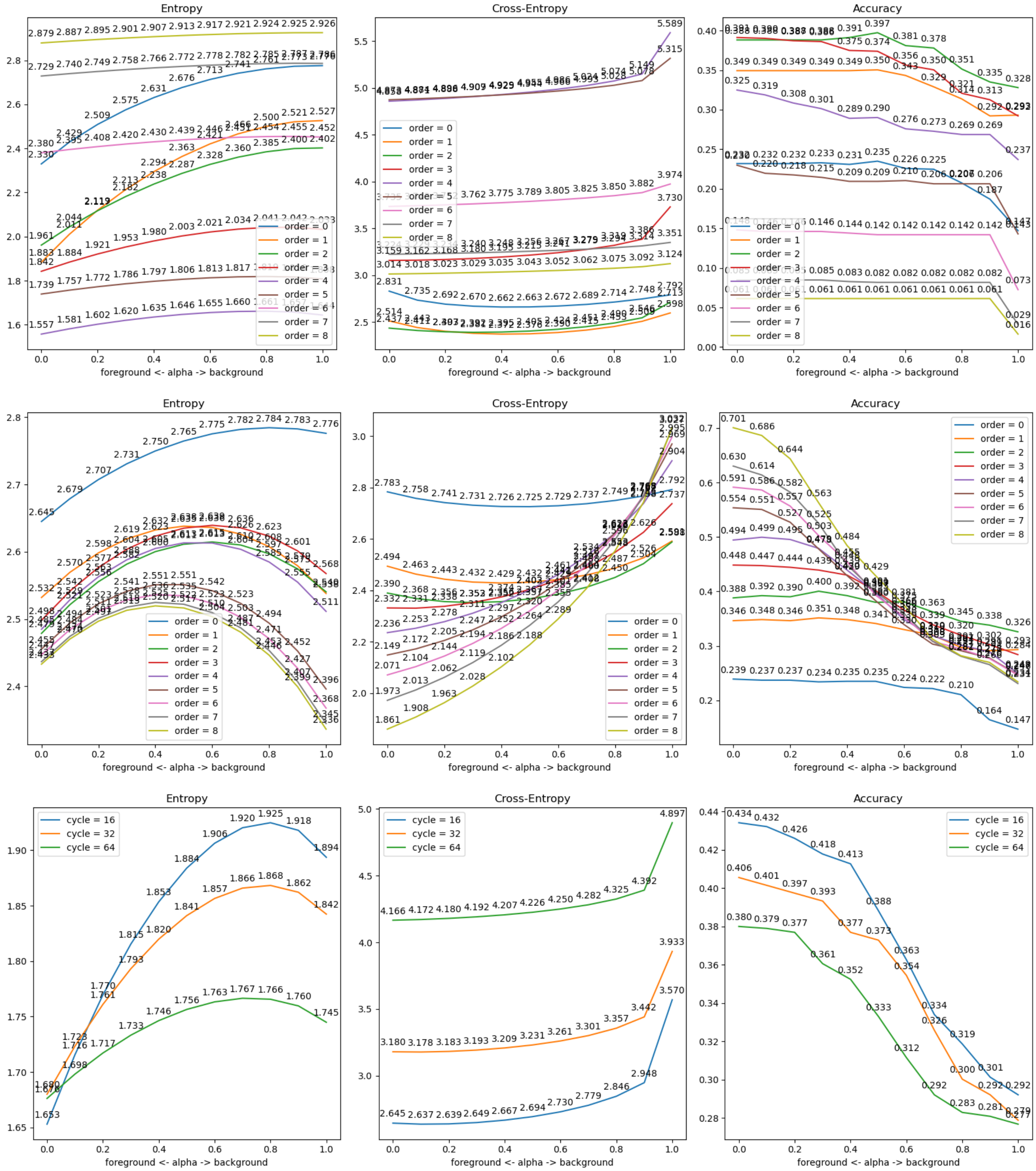

Figure 4.2: Entropy, cross-entropy and accuracy metric of Markov model, variable-order Markov model, and bar-cycle model on PDSA dataset.

types of note in total.

4.3 Prediction result of the single models

We used 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 Figure 4.1 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, which means that the melody sequences are more predictable under the context of the same song rather than the whole dataset.

In the following sections, we are going to analyze the performance of the variable-order Markov model and the bar-cycle model in detail.

4.3.1 Prediction result analysis of the variable-order Markov model

The prediction accuracy of the 8th-order foreground variable-order Markov model reaches 92% on the POP909 dataset, the cross entropy gets to lower than one and the entropy gets to 1.454, 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. Due to the variable-order Markov model is a mixture of different order Markov model, it also outperformed the standard Markov models.

The order that the true term hit in the foreground variable-order Markov is as in the following chart:

(chart)

We can see from the chart that the order is almost evenly distributed over order 2 to order 8, indicating that repetitions of different length happen pervasively throughout the song.

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

Success cases: *To be done*

Failed cases: *To be done*

4.3.2 Prediction result analysis of the bar-cycle model

Consider the

Success cases: *To be done*

Failed cases: *To be done*

4.4 Combining bar-cycle model with variable-order Markov model

We can see from the previous analysis that these two different approaches actually represent different aspects of repetition behavior of music. Then another question will be whether there are any ensemble methods to further decrease the error rate based on the results of the two models.

Here, we selected *, *, * of the variable-order Markov model and the entropy and confidence of the distributions generated from the two models as features to predict which distribution should we choose out of the two models. This is a classical binary classification problem and can be solved by an SVM model.

Chapter 5

Conclusion