New evaluation methods for automatic music generation

Zongyu Yin

PhD

University of York

Computer Science

May 2022

Abstract

Recent research in the field of automatic music generation lacks rigorous and comprehensive evaluation methods, creating plagiarism risks and partial understandings of generation performance. To contribute to evaluation methodology in this field, I first introduce the originality report for measuring the extent to which an algorithm copies from the input music. It starts with constructing a baseline to determine the extent to which human composers borrow from themselves and each other in some existing music corpus. I then apply the similar analysis to musical outputs of runs of MAIA Markov and Music Transformer generation algorithms, and compare the results to the baseline. Results indicate that the originality of Music Transfomer's output is below the 95% confidence interval of the baseline, while MAIA Markov stays within that interval.

Second, I conduct a listening study to comparatively evaluate music generation systems along six musical dimensions: stylistic success, aesthetic pleasure, repetition or self-reference, melody, harmony, and rhythm. A range of models are used to generate 30-second excerpts in the style of Classical string quartets and classical piano improvisations. Fifty participants with relatively high musical knowledge rate unlabelled samples of computer-generated and human-composed excerpts. I use non-parametric Bayesian hypothesis testing to interpret the results. The results show that the strongest deep learning method, Music Transformer, has equivalent performance to a non-deep learning method, MAIA Markov, and there still remains a significant gap between any algorithmic method and human-composed excerpts.

Third, I introduce six musical features: statistical complexity, transitional complexity, arc score, tonality ambiguity, time intervals and onset jitters to investigate correlations to the collected ratings. The result shows human composed music remains at the same level of statistical complexity, while the computer-generated excerpts have either lower or higher statistical complexity and receive lower ratings.

This thesis contributes to the evaluation methodology of automatic music generation by filling the gap of originality report, comparative evaluation and musicological analysis.

Acknowledgements

Completing a Ph.D. is never easy. Apart from gaining knowledge, this experience has inspired me to understand and explore the world and myself. This spiritually long journey would not be complete without the support I received.

I would first like to thank my supervisor Tom Collins, whose expertise and patience broadened my horizons, bring my research work to another level, and enabled me to learn from it and become a qualified scholar.

I would also like to thank my other two supervisors Susan Stepney and Federico Reuben. Not only have you supported me with my research work, but you have given me the strength and motivation to stay positive and persevere.

I would like to thank my parents, especially my mother Yinghong Xiao, for always supporting my life decisions and letting me choose who I want to be.

In addition, I would like to thank my best pal Hao Sun, I cannot go a day without your afternoon teas.

Zongyu Yin,
London,
Apr 2022.

Declaration

I declare that this thesis is a presentation of original work and I am the sole author. This work has not previously been presented for an award at this, or any other, University. All sources are acknowledged as References. Publications arising from the thesis are listed:

Chapter 6 is based on the following conference paper.

> Zongyu Yin, Federico Reuben, Susan Stepney and Tom Collins. "A Good Algorithm Does Not Steal–It Imitates": The Originality Report as a Means of Measuring When a Music Generation Algorithm Copies Too Much. *International Conference on Computational Intelligence in Music, Sound, Art and Design (Part of EvoStar)*, pp. 360-375, 2021.

Chapter 6 is also based on the following journal paper (an invited extended version of the above conference paper).

> Zongyu Yin, Federico Reuben, Susan Stepney and Tom Collins. Measuring When a Music Generation Algorithm Copies Too Much: The Originality Report, Cardinality Score, and Symbolic Fingerprinting by Geometric Hashing. *SN Computer Science, Springer Nature*, pp. 1-18, 2022.

Chapter 7 is based on the following journal paper that has been under review.

> Zongyu Yin, Federico Reuben, Susan Stepney and Tom Collins. Deep learning's shallow gains: A comparative evaluation of algorithms for automatic music generation. Submitted to *Machine Learning, Springer*, 2022.

Contents

List of Tables

List of Figures

1 Introduction

Automatic music generation (AMG) uses computational methods to create music (Carnovalini and Rodà 2020), such methods endow machines with computational creativity that enables creative behaviours. AMG, as one of the creative behaviours, has long been explored along the development of machine learning approaches. Because of the nature of multidisciplinary research field, an absolute definition of creativity does not exist, Jordanous and Keller (2016) apply methods of natural language processing to identify keywords that can be used to describe "creativity". Boden et al. (2004) defines creativity as a process of producing novel and valuable ideas. Specifically, the novelty was further discussed in both psychological and historical aspects: the psychological novelty means the generated idea is new to the individual; the historical novelty is meanwhile psychological and never existed in history as well. Most systems of computational creativity aim for psychological novelty. This descriptive concept was further developed by Wiggins (2006) to reinforce the boundary of systems being creative. The work by Boden et al. (2004) has been considered a valuable attempt not only by bridging artificial intelligence with cognitive science but also to reveal the feasibility of computer creativity despite the doubt and denial from other research (e.g., Jefferson 1949). As for practical use cases, AMG systems have been

widely used to create functional music (e.g. advertising music), and often as a creative tool for educational purposes and to inspire musicians.

This thesis aims to contribute to the evaluation methodology of AMG, particularly in the aspects of originality analysis and musicological interpretation. As mentioned, creativity is an abstract object and difficult to measure directly, my work here revolves around more concrete features extracted from output and analyses them separately. To cover the most common description of creativity, my evaluation of AMG systems measures relative originality and selected musical dimensions (e.g., stylistic success, repetitive structure, melody). Chapter 5 explicitly states the research questions that are addressed in this thesis. Briefly, the first question is to what extent the current models generate original music compared to the human composer baseline; the second question is about the musicological quality gap between the music generated by deep and non-deep learning systems and human composers; the third question is what musical features can be extracted from music data and represent the quality of each musical dimension. Chapter 5 also describe how these three questions are addressed in the technical Chapters 6, 7 and 8.

1.1 Originality

A quotation from Igor Stravinsky reads: "A good composer does not imitate, he steals" (Yates 1968). The quotation, while made in relation to a serial work, reflects Stravinsky's general interest in incorporating melodies, harmonic language, and forms from previous periods into new works such as his *Pulcinella Suite* (1922). Stravinsky uses the term "imitate" with a negative connotation: he would rather steal, say, a melody wholesale and rework

it in a contemporary piece, than he would make mere allusions to (imitate) the work of past or contemporary composers. With respect to the current paper's context – the rise of AI music generation algorithms – I instead use the term "imitate" with a positive connotation and the term "steal" with a negative connotation. As I show in Chapter 6, some deep learning algorithms for music generation (Huang et al. 2018) are copying chunks of original input material in their output, and I would count it as a success if an algorithm – from the deep learning literature or otherwise – could generate output that *sounds like* (imitates) – but does not *copy from* (steal) – pieces in a specific style. Research on artificial intelligence has achieved various feats of simulating human perception (e.g., Graves et al. 2013; Zhang et al. 2015; He et al. 2016) and production (e.g., Radford et al. 2015; van den Oord et al. 2016; Devlin et al. 2018; Brown et al. 2020). A number of music generation models have been developed in recent decades, many predating or outside of deep learning (Todd 1989; Collins and Laney 2017) and some espousing a belief in the superiority of deep learning (Roberts et al. 2018; Donahue et al. 2019). I have observed, with increasing alarm, that deep learning papers on music generation tend to rely solely or primarily on loss and accuracy as a means of evaluation (Huang et al. 2018; Roberts et al. 2018). If there are listening studies, they employ listeners with inadequate expertise, and there is little or no musicological analysis of outputs, and no analysis of whether generated material plagiarises (steals from) the training data. As an increasing number of musicians are now incorporating AI into their creative workflows, checking an AI's output for plagiarism is now a paramount challenge in this area. To this end, this work considers the topic of **automatic stylistic composition**

– a branch of AMG where there is a stated stylistic aim with regards to the algorithm output, and a corpus of existing pieces in the target style.

Chapter 6 addresses research question 1: "to what extent does the AMG system produce original output?" I first aim to establish a framework for checking the originality of auto-generated music with a specified style. I introduce and exemplify the originality report as a means of measuring when a music generation algorithm copies too much. I discuss how to calculate a distribution for the extent to which human composers borrow from themselves or each other in some corpus of pieces in a specific style; then I discuss how to use this as a baseline while moving a sliding window across a generated passage and measuring originality as a function of time in the generated material. The originality report is parameterised by a "similarity score", so the framework is adaptable to measures that are more appropriate to certain characteristics of different datasets. I demonstrate the originality report with case studies of different similarity scores: the cardinality score and the fingerprinting score. The two measurement methods are used on different datasets, which differ mainly in terms of containing expressive timing data or not. The report is complemented by a musicological analysis of outputs from prominent deep (Huang et al. 2018) and non-deep (Collins and Laney 2017) learning models. For the deep learning model, I also investigate how originality varies with training epochs.

1.2 Stylistic success and other musical dimensions

In the past decade, breakthroughs in artificial intelligence and deep learning have been established as such through rigorous, comparative evaluations[1], for example, in computer vision (O'Mahony et al. 2019) and automatic speech recognition (Toshniwal et al. 2018). In the field of AMG, however, to my knowledge, there has been no comparative evaluation to date between deep learning and other methods (Huang et al. 2018; Yang et al. 2017; Dong et al. 2018; Hadjeres et al. 2017; Thickstun et al. 2019; Donahue et al. 2019; Tan and Herremans 2020).[2] Rather, it appears to have been assumed that deep learning algorithms must have similarly superior performance on AMG. Thus, Chapter 7 concerns research question 2:

1. Is deep learning superior to other methods on the task of generating stylistically successful music?[3]

2. Are any computational methods approaching or superior to human abilities on this task?

In recent decades, several methodologies have been applied to tackle mu-

[1]The term "comparative evaluation" refers to a scenario in which two or more systems are compared with respect to their performance on a well-defined task. The purpose of evaluation is to extend knowledge by improving understanding of how systems or algorithms perform under certain conditions, and whether one method or another is superior to human abilities on a task and/or compared to other computational approaches. Additional benefits of evaluation include determining the feasibility of methods and providing a basis for characterising and improving upon systems in future work.

[2]Examples of recent evaluations include (Yang and Lerch 2020; Sturm et al. 2019a; Janssen et al. 2019), but none constitute direct comparisons between deep learning and other methods.

[3]A "stylistically successful" excerpt can be defined as one that conforms, in a listener's opinion, to the characteristics of some target style.

sic generation tasks, and these methods can be categorised by two musical data representations: raw audio (Mehri et al. 2017; van den Oord et al. 2016) and symbolic tokens (Thickstun et al. 2019; Roberts et al. 2018; Collins and Laney 2017; Huang et al. 2018). Here, I focus on symbolic methods for generating polyphonic music.[4] Recent deep learning-based systems are claimed, by their authors, to display state-of-the-art performance, but this is only in comparison with earlier deep learning-based systems (e.g., Huang et al. 2018; Yang et al. 2017; Dong et al. 2018; Hadjeres et al. 2017; Thickstun et al. 2019; Donahue et al. 2019; Tan and Herremans 2020).[5] The consequence is an echo chamber, where deep learning for AMG is evaluated in isolation from other methods, yet the corresponding papers claim state-of-the-art performance. Here I describe a comparative evaluation across a broader range of music generation algorithms, which enables me to address the question "Are deep learning methods state-of-the-art in the automatic generation of music?"

Evaluation by participants of appropriate expertise,[6] when conducted and analysed in a rigorous manner with respect to research design and statistical methods, has long been considered a strong approach to evaluating generative (music) systems (Ariza 2009), because it has the potential to reveal the effect of musical characteristics in a system's output on human perception, and it models the way in which student stylistic compositions have been evaluated

[4]The term polyphonic refers to music where more than one note may begin or be sustained while others begin or are sustained.

[5]The term state-of-the-art has also been used rather freely in these papers. I argue it should be reserved for describing a system whose performance is statistically significantly better than others, according to the results of a listening study where relevant existing systems encompassing diverse approaches are compared, and the description of the method – especially recruitment and musical background of participants – is detailed enough to enable replication.

[6]Appropriate expertise can be defined as being familiar with a style of music, but not knowing it so well that one can explicitly recognise excerpts as being from specific pieces.

in academia for centuries (Collins et al. 2016b). An alternative to evaluation by listeners is to use metrics such as cross-entropy and predictive accuracy (Huang et al. 2018; Hadjeres and Nielsen 2020; Johnson 2017; Thickstun et al. 2019), or distributions of automatically calculated musical features (e.g., pitch class, duration (Yang and Lerch 2020)), and investigate how such features differ, say, between training data and system output. The automaticity and speed of evaluation by metrics are major advantages, but evaluation by metrics presupposes that the metrics are accurate proxies for the complex construct of music-stylistic success or other musical dimensions. If I knew how to define music-stylistic success as a set of metrics, it would be of great help in solving the challenge of AMG, because the objective function for the system could be obtained and it would be possible to generate music that scored highly according to that definition.

My review of existing approaches to evaluation finds that the musical dimensions tested in listening studies often vary according to research interests, and so are inconsistent. The performance of deep learning-based systems is often evaluated with loss and accuracy, which do not reflect the stylistic success (or other musical dimensions) of algorithm output. Different evaluations' foci make comparison between models difficult. I argue that although the use of metrics is necessary, it is not sufficient for the evaluation of computer-generated music. Here I address the question "What does the generated music sound like to human listeners of an appropriate level of expertise?" In the listening study (see Chapter 7), the performance of four machine learning models is assessed directly by human perception, which is represented by the rating of six musical dimensions. These musical dimensions are derived

from previous analyses of classical music (Rosen 1997): *stylistic success* and *aesthetic pleasure* (Collins et al. 2016b; Collins and Laney 2017), *repetition*, *melody*, *harmony* and *rhythm* (Hevner 1936), defined in Section 7.4.1.

I apply non-parametric Bayesian hypothesis testing (van Doorn et al. 2020) to the ratings collected from the listening study, to verify hypotheses about differences in performance between systems. The Bayesian hypothesis test is a test between two mutually exclusive outcomes. It allows for the possibility of finding a statistically meaningful *non*-difference in performance between systems; in contrast, the standard frequentist hypothesis testing framework can only fail to reject a null hypothesis of no difference between systems, which is unsatisfactory because this result can also be due to an under-powered test (a more detailed explanation is given in Section 4.3). The conclusions that can be drawn from Bayesian hypothesis tests are also complementary and arguably preferable to just describing and displaying statistical features of systems (Yang and Lerch 2020).

Finding the correlation between the mentioned musical dimensions and musical features provides the connection between subjective and objective evaluation of music. An ideal feature predicts human's judge in a certain aspect of music, as features can be extracted from music, which saves time and effort in conducting the listening study. However, musical features in existing work tend to be fundamental and low-level, as they are often statistics calculated based on a single factor (e.g., pitch class). Such musical features have been used to show characteristics of corpus (Sturm and Ben-Tal 2017; Dong et al. 2018; Yang and Lerch 2020), but more advanced features and their correlation to subjective ratings have been rarely studied. Chapter 8

aims to address research question 3: "What musical features can be extracted from music data to represent the quality of various musical dimensions?" I present several higher-level features and investigate their correlation to the ratings collected in Chapter 7.

The following Chapters 2, 3 and 4 form the literature review. Chapter 2 provides a preliminary introduction to music representations, from Western musical notation and its digital formats to representations that are closer to model input. As data representation often interacts with model architecture, it is recommended to understand these concepts before reading the technical chapters that follow. Chapter 3 reviews AMG systems. Depending on the underlying generation method, they are classified into rule-based approaches (Ebcioğlu 1990; Bel and Kippen 1992; Anders and Miranda 2010; Quick and Hudak 2013), Markovian sequential models (Cope 1996; Allan and Williams 2005; Eigenfeldt and Pasquier 2010; Collins and Laney 2017; Herremans and Chew 2017), artificial neural networks (Todd 1989; Mozer 1994; Hild et al. 1991) and deep learning methods (Oore et al. 2018; Huang et al. 2018; Roberts et al. 2018; Thickstun et al. 2019; Dong et al. 2018). Chapter 4 focuses on evaluation methods for creative systems, especially music generation. The evaluation framework and experimental design in Chapter 7, are largely inspired by these reviewed endeavours.

Even though the main contribution of this thesis has been mentioned more or less in this introduction, I explicitly formalise the research questions in Chapter 5, and how they are addressed in Chapters 6, 7 and 8.

2

Literature review: Music representations

This chapter mainly serves to describe the development of music representations. It starts with Western musical notation, which is the nearest original form of the music presented by composers for the majority of dataset mentioned in this thesis. The principle of the review of stave notation is to establish an understanding of the basics of musical elements within a piece, so as to understand their structures and attributes, without touching more advanced music theory knowledge. Next, I introduce the three widely used encoding methods for restoring and transmitting music. This is necessary because the AMG models, that are involved in experiments of this thesis, require an extra process to translate the raw output into one of those file formats, so that the generated music can be sounded. Finally, the music representations for statistical music analysis and AMG can be varied depending on the research interests or constraints at the implementation level. It is necessary to introduce the representation methods, since these topics are frequently discussed and exploited in subsequent technical parts.

2.1 Musical notation

Musical notation is any system that utilises various types of symbols to indicate musical elements including the presence/absence of sound, speed and rhythm patterns, so that music can be printed, read and performed. Its format and ingredients often vary geographically and historically. Stave notation is one of the most widely used musical notation nowadays, it consists of a set of five horizontal lines and four spaces, each of which denotes a different musical pitch. Considering piano as an example, the pitch of keys is usually distributed according to the chromatic scale, where an octave is equally spaced by twelve pitches or semitones, and increasing an octave is a doubling of frequency. Figure 2.1 shows pitch names of lines and spaces on the stave and piano keys within an octave.

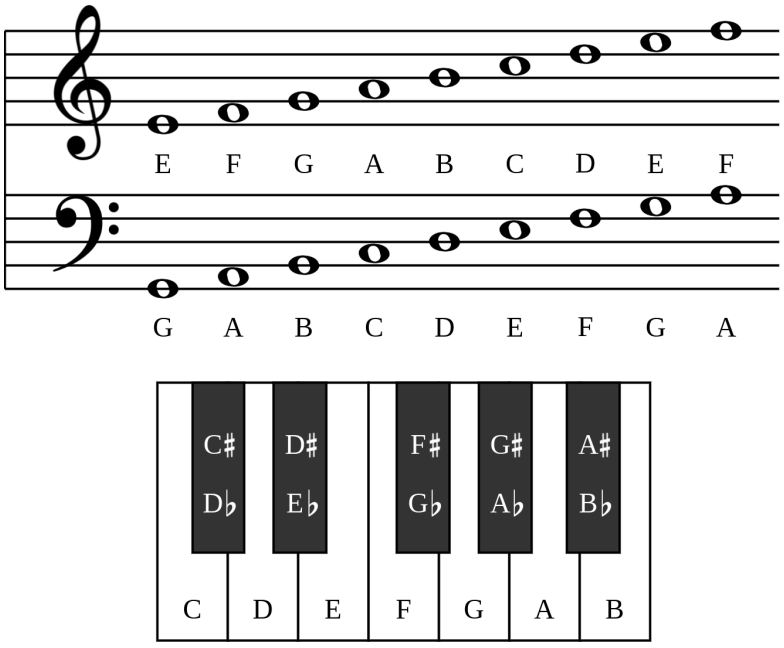


Figure 2.1: The stave at the top shows the pitch name of lines and spaces for both treble and bass staves, and the keyboard at the bottom shows the pitch name of piano keys within an octave.

A set of pitches with the same name across different octaves is called a pitch class, usually an additional number is added after the pitch name to locate the exact pitch or key on the keyboard, for example, C5 means the C at the fifth octave, which is the same C notated in the treble staff in Figure 2.1. There are already some symbols mentioned but not yet explained, Figure 2.2 provides a limited set of primary symbols and their corresponding name.

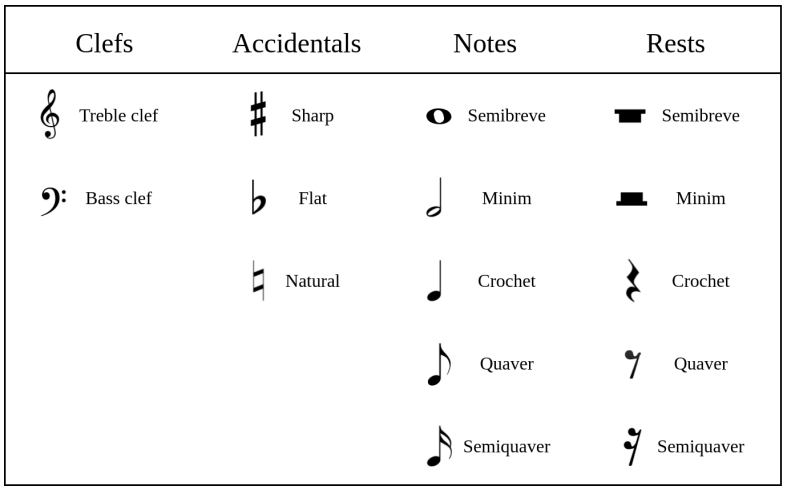


Figure 2.2: A limited set of symbols can be commonly found in stave notation.

A stave notated with a treble or bass clef maps different pitches to lines and spaces, which is already demonstrated in Figure 2.1. A sharp/flat accidental raises/lowers the pitch by a semitone, while a natural accidental cancels previous accidentals. The symbols of both notes and rests are determined by the relative duration. The note/rest value is a fractional power of two, for example, a minim is half long of a semibreve, a crochet is half long of a minim, and a quaver is half long of a crochet. Combining all the elements above, Figure 2.3 shows example measures written in stave notation. The same piece is used in later sections to demonstrate the conversion across different representations.

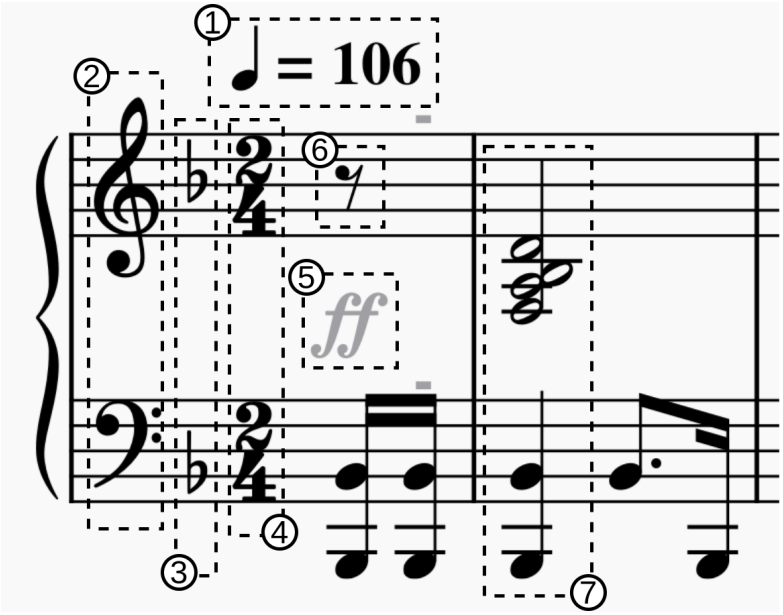


Figure 2.3: Example measures written in stave notation.

The explanations of the symbols in Figure 2.3 are linked by the attached indices:

1. The tempo of the piece is usually notated at the beginning of a piece, it measures the speed in beats per minute (bpm). The example indicates 106 crochet beats per minute.

2. The treble clef (located above) and bass clef (located below) indicate what pitches are represented by the lines and spaces on the musical stave. There are other clefs except for treble and bass, such as alto clef being most often used for viola music.

3. The key signature denotes the major or minor key by a set of ♯ or ♭ symbols placed at the specific lines and spaces on the stave. The example with a single ♭ on the pitch of B is D minor.

4. The time signature denotes the note value that is equivalent to a beat (the number 4 below) and the number of beats per measure (the number 2 above), where a measure is an area enclosed by two adjacent bars. Thus, the time signature of the example indicates each measure should last a minim long in total, the incomplete first measure, with only a quaver long, is called an anacrusis.

5. The dynamics symbol indicates the relative variation in loudness for the performers, which is one of the expressive elements in music performance. The example dynamics symbol, fortissimo, stands for "very loud".

6. The rest symbol denotes the absence of sound with a certain duration, which is a quaver long for the example.

7. The musical notes denote the presence of sound with pitch and duration, multiple notes played simultaneously to form a chord, which is notated as note heads attached to the same beam.

With regard to the conversion between canonical durations (e.g., minim, crochet and quaver) and time in seconds can be expressed as:

$$s = \frac{d * 4}{bpm/60} \tag{2.1}$$

where s denotes the time in seconds and d denotes the canonical durations: 1/2 for minim, 1/4 for crochet and 1/8 for quaver.

2.2 Digitisation in music

After entering the information age, the digitisation of music has emerged. This section does not consider the music with its raw audio recorded in or converted to a digital format, such as the compact disc, MP3 and other compressed music formats, while it focuses on the digital formats that retain the symbolic form of music. The following subsections introduce three formats: musical instrument digital interface, MusicXML and kern. These formats are widely exploited to maintain the dataset in the area of AMG and statistical analysis of music.

2.2.1 Musical instrument digital interface

Musical instrument digital interface (MIDI) is introduced by Smith and Wood (1981), as a communication protocol that allows digital instruments (e.g., synthesiser, electronic piano and drum kit) to connect with each other and computers. Instead of manipulating audio, MIDI utilises "messages" to enable the playing, recording and editing of music. The MIDI messages do not carry the sound itself, but only the sequence of instructions to create the sound. Generally, a MIDI message is transmitted as multiple bytes in a time sequence. The first byte is a status byte determining the types of messages, which can be categorised into channel messages and system messages. The primary note-on and note-off messages are channel-specific messages, where the second byte determines the MIDI note number as pitch ranging from 0 to 127 (88 keys on a piano are mapped from 22 to 108), and the third byte determines the velocity value also ranging from 0 to 127. Velocity in MIDI is the force with which a note is played, so it describes the loudness or dynam-

ics of a note instead of speed in the common case. When a key is pressed on an electronic piano, a note-on message with the corresponding MIDI note number and velocity is sent and registered to the computer, and when this key is released, a note-off message is sent. Moreover, the non-channel specific system real-time messages are used for time synchronisation. Although the messages are registered and sequenced in real-time, the base unit for time is "tick", every MIDI file has to specify ticks per quarter note (crochet) and beats per minute, so that the ticks can be transferred into seconds with:

$$s = t * tpq * bpm * 60$$

where s is the time in seconds, t is the number of ticks, tpq denotes ticks per quarter note and bpm denotes beats per minute. The term "resolution" often refers to ticks per quarter notes, a low resolution can result in the recorded expressive performances to sound metronomic or robotic, because the subtle variations in timing are eliminated by the quantisation.

Generally, the pieces written in the above stave notation can be converted into MIDI file format with message sequences. The sequence is usually visualised with piano roll notation, a piano roll is very much similar to a stave where the horizontal axis represents time and the vertical axis represents pitch. Figure 2.4 shows a piano roll containing the same set of notes described in Figure 2.3.

Figure 2.4: The example piece in piano roll representation.

Piano rolls use bars to denote the presence of musical notes. The time is equally divided by the vertical lines, and the duration of a note is represented by the length of the bar. The pitch is mapped as MIDI note numbers from bottom to top in ascending order.

2.2.2 MusicXML

In contrast to MIDI which aims to provide a universal communication protocol, MusicXML is exploited as an XML-based interchange file format for representing sheet music specifically, and it is widely supported by notation programs like MuseScore. Similar to any other markup languages (e.g., HTML and XML), MusicXML utilises tags to notate musical attributes and form a hierarchical structure. For example, specifying a semibreve note with the pitch of C5 results in the following code:

```
1     <note>
2       <pitch>
3         <step>C</step>
4           <octave>5</octave>
```

```
5           </pitch>
6           <duration>4</duration>
7           <type>whole</type>
8           </note>
```

The whole set of MusicXML code for the example piece shown in Figure 2.3 can be accessed in Appendix A.1, where more tags like "measure", "staff" and "chord" are demonstrated to constitute a more complete piece.

2.2.3 Kern

Kern is another encoding method to represent Western music, similar to MusicXML, various types of entities are used to provide a human-readable digital format, but kern arranges notes in a pure sequential form rather than the hierarchical nature embedded in MusicXML. The following kern code represents the same piece in Figure 2.3

```
1     **kern                **kern
2     *staff2               *staff1
3     *clefF4               *clefG2
4     *k[b-]                *k[b-]
5     *M2/4                 *M2/4
6     16BBB-LL 16BB-        8r
7     16BBB-JJ 16BB-        .
8     =1                    =1
9     4BBB- 4BB-            2F 2A 2B- 2d
10    8.BB-L                .
11    16BBB-Jk              .
12    *-                    *-
```

The two columns of code correspond to the two staves, which is extendable, a string quartet can have four columns. Each column starts with "**kern" at line 1, and the following attributes, including the stave number, clef, key signature and time signature, are specified in lines 2-5 starting with "*". With the syntax of kern, C4 is described as a single lower-case

letter “c”. The octaves above the fourth are represented by repeating the same letter, for example, C5 is represented by cc, C6 by ccc and so on. For octaves below the fourth, upper-case letters are used, C3 is represented by C, C2 by CC and so on. The scheme of repetition also applies to other pitch names. The sharp/flat symbol is represented by “#”/“-” appended after the pitch. Rest tokens are denoted by the lower-case letter r along with a duration appended at the front. The canonical durations, such as crochet, quaver and semiquaver, are represented by the denominator of the duration fraction: 4 for crochet, 8 for quaver, 16 for semiquaver and so on. Considering the example piece, the first two-note chord of the anacrusis is specified as “16BBB-LL 16BB-”, where “LL” denotes the beginning of a beam and “JJ” at line 7 encloses the beam. A beam is a horizontal bar that connects consecutive notes to indicate rhythmic grouping. “=1” denotes the start of the first measure, and “*-” ends the whole piece.

2.3 Symbolic representation

The previous section describes three formats that have been widely used for encoding music in its symbolic form. However, training music generation models and conducting statistical analysis often require the data in a more monotonous and specific form. The following subsections provide the review of forms designed for various research works, the mentioned models are discussed in detail in Chapter 3.

2.3.1 Derivation of piano roll

Regarding the piano roll mentioned in Figure 2.4, it can be naturally converted into a matrix:

$$
\begin{array}{c}
127 \\ \vdots \\ 46 \\ 45 \\ 44 \\ 43 \\ 42 \\ 41 \\ 40 \\ 39 \\ 38 \\ 37 \\ 36 \\ 35 \\ 34 \\ \vdots \\ 0
\end{array}
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\begin{array}{c}
\\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\
\end{array}
$$
$$
\begin{array}{cccccccccc}
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9
\end{array}
$$

which have the size of 128×10, corresponding to 128 MIDI note numbers and equally spaced time steps of a semiquaver. As the limited space, it only contains the octave chords located at the lower stave in Figure 2.3. With the presence of notes being denoted as **1**, the whole matrix can be regarded as a binary image. Yang et al. (2017) and Dong et al. (2018) introduce generative adversarial networks trained with this image-like music data for music generation.

Alternatively, each column of the matrix can be regarded as a one-hot or many-hot vector. One-hot encoding is a method often used to map categori-

cal variables (e.g., MIDI note number) into a vector with the size of possible elements, where the corresponding element of the encoding vector is set to 1 and all other elements to 0, while the many-hot encoding allows multiple elements of the vector is set to 1. Thus, steps 6-8 in the matrix represent one-hot vectors for the MIDI note number 46, and the other steps represent many-hot vectors for the MIDI note numbers 46 and 34. One-hot encoding can be used for representing monophonic melodies where only one note at a time, and many-hot for polyphony which contains simultaneous notes (e.g., chords). But the described encoding methods are vulnerable, for example, holding a single note and repeating consecutive same-pitched notes can not be distinguished based on the representations. Roberts et al. (2018) use multiple one-hot vectors to represent MIDI data with multiple tracks, where each track is monophonic and the vector size is extended from 128 (corresponding to MIDI note numbers) to 130, where the two extra tokens denote "note-off" and "rest". The "note-off" token indicates the end or offset of the previous note, and the silent step is explicitly indicated by "rest".

2.3.2 Event-based serialisation

As mentioned previously, one/many-hot encoding does not show a good adaptation of representing polyphony, expressive timing and dynamics. Oore et al. (2018) introduce a novel serialisation for encoding polyphonic music with expressive timing and dynamics. This method shares a close concept with MIDI messages, as it contains four types of events:

- 128 **NOTE_ON** events corresponding to MIDI note numbers, each of them starts a note with its associated MIDI note number.

- 128 **NOTE_OFF** events corresponding to MIDI note numbers, each of them ends a note with its associated MIDI note number.
- 100 **TIME_SHIFT** events representing multiples of 10 milliseconds up to 1 second, each of them moves the time step forward by its associated duration, TIME_SHIFT events can be stacked to represent a duration longer than a second, for example, 2.5 seconds is represented as a sequence of TIME_SHIFT_1000, TIME_SHIFT_1000, TIME_SHIFT_500.[1]
- 32 **SET_VELOCITY** events obtained by binning the original 128 MIDI velocity values, according to the author, the binning is made as a human can hardly perceive the minor difference of too fine velocity granularity. A SET_VELOCITY event set the velocity value for the all following notes until a new SET_VELOCITY event is specified.

Again, applying this serialisation method to the piece shown in Figure 2.3, with the assumption of every note having the MIDI velocity value 112, results the following events sequence:

```
SET_VELOCITY_28
NOTE_ON_34    NOTE_ON_46    TIME_SHIFT_140
NOTE_OFF_34   NOTE_OFF_46
NOTE_ON_34    NOTE_ON_46    TIME_SHIFT_140
NOTE_OFF_34   NOTE_OFF_46
NOTE_ON_34    NOTE_ON_46
NOTE_ON_65    NOTE_ON_69    NOTE_ON_70    NOTE_ON_74    TIME_SHIFT_570
NOTE_OFF_34   NOTE_OFF_46   NOTE_ON_46    TIME_SHIFT_420
NOTE_ON_34    TIME_SHIFT_140
NOTE_OFF_34   NOTE_OFF_65   NOTE_OFF_69   NOTE_OFF_70   NOTE_OFF_74
```

Figure 2.5: The example piece represented by the event-based serialisation.

[1] Although the original paper uses 125 TIME_SHIFT events as multiples of 8 milliseconds, this thesis adhere to the configuration described by Huang et al. (2018) for unifying the representation.

where the time in milliseconds is calculated according to Equation 2.1 and rounded to multiples of 10 milliseconds, event tokens are arranged only for ease of visually corresponding to notes. Oore et al. (2018) take this serialisation method to train recurrent neural networks for music generation, Huang et al. (2018) adopt the same method to train transformer models which achieves higher performance (lower validation loss) than the recurrent neural networks. MuseNet by OpenAI[2] extended this serialisation with extra events indicating instruments and composers, so that the model can generate music with 10 different instruments and allow style mixing between composers.

Compared with one/many-hot encoding, event-based serialisation has its advantages in adapting expressive timing and dynamics, and can be extended for other attributes like instruments and composers. It does not mean that these elements or attributes cannot be represented by using one/many-hot encoding, but it can dramatically increase the size of dimensions yet the data remain sparse, that is, the number of 1s is extremely lower than the number of 0s. However, there are drawbacks to using this serialisation. During the generation process in an autoregressive manner, it is not guaranteed that a NOTE_ON is paired with a NOTE_OFF later, a falsely produced TIME_SHIFT event can make all following notes offbeat.

2.3.3 Geometric form

The piano roll representation is introduced in the above section, instead of further formulating it into a sequence of one/many-hot vectors, alterna-

[2] `https://openai.com/blog/musenet/`

tively, it can be considered in a geometric manner, where a set of points representing the notes is attributed by MIDI note number and onset in the two-dimensional space. A note can also have more than two attributes, such as velocity and duration. Meredith et al. (2002) encode a piece into a finite set of tuples/vectors:

$$D = \{d_1, d_2, \ldots, d_n\}$$

where n is the number of notes and each tuple d specifies the note with five attributes: onset time counted in semiquavers, chromatic pitch (MIDI note number), morphetic pitch (Meredith 1999), duration and voice index. The piece in Figure 2.3 is then represented as a set of tuples:

$$\{\{0, 34, 45, 1, 2\}, \{0, 46, 52, 1, 2\}, \{1, 34, 45, 1, 2\},$$
$$\{1, 46, 52, 1, 2\}, \{2, 34, 45, 4, 2\}, \{2, 46, 52, 4, 2\},$$
$$\{2, 53, 56, 8, 1\}, \{2, 57, 58, 8, 1\}, \{2, 58, 59, 8, 1\},$$
$$\{2, 62, 61, 8, 1\}, \{6, 46, 52, 3, 2\}, \{9, 34, 45, 1, 2\}\}$$

Music in the set of points is largely exploited for feature extraction and statistical analysis described in Chapter 8, as the onset time is explicated specified inside the tuple, the order of points does not affect the result of many tasks, such as calculating the mean and variance of pitches. But the order can matter for tasks like calculating inter-onset-intervals, which is discussed in Section 8.1.5. So, without further specifying, the set of points discussed in later technical sections is sorted in ascending onset time and pitches.

Although many music generation models discussed in Chapter 3 consider music in sequential form (e.g., one/many-hot vectors and event tokens),

Jeong et al. (2019b) represent a geometric form of music for training note-level gated graph neural network, as a component of measure-level hierarchical attention recurrent neural network, which can render expressive piano performance from the music score. In this geometric form, notes are represented as nodes in a graph where each node is connected with neighbours, by six types of edges: next, voice, slur, onset, sustain and rest.

Generally, researchers take **sequential** (e.g., Conklin 2010; Conklin and Witten 1995) or **geometric** (e.g., Meredith et al. 2002; Collins et al. 2016a) approaches to the representation and comparison of music. There are pros and cons to each approach. With the sequential approach, if one chooses to focus on MIDI note numbers alone and two melodies have the same MIDI notes (up to transposition) but different rhythms, a sequential representation (specifically, difference calculations between consecutive notes) will recognise these melodies as similar, whereas a geometric representation may not. However, with a sequential representation, it is less obvious how to handle polyphony, whereas a geometric representation can encode a polyphonic piece as easily as it encodes a monophonic piece. For instance, in the sequential representation shown in Figure 2.5, the tokens encoding the occurrence of F3 in the higher stave and last B♭1 in the lower stave are eight indices apart, even though the notes sound together. So any parameter that allows these events to be recognised as related has to be large enough to span this gap in indices. Moreover, an embellished (or, on the other hand, reduced) variation of some melody may not be recognised by the sequential representation as similar because the relationships between adjacent notes will be altered by the added or removed notes, even though the "melodic scaffold"

remains intact. A geometric representation may be more robust to this kind of variation.

2.4 Summary

This chapter starts with an introduction to Western musical notation and uses the same short phrase to show how it can be transformed into digital formats, including MIDI, MusicXML and Kern. Finally, I describe the data formats that are more directly used for music information retrieval tasks, including piano roll representation, event-based serialisation and geometric form. AMG systems typically use these data formats depending on the generation methods. This leads to the next chapter, which reviews AMG systems by categorising their methods.

3

Literature review: Automatic music generation

This chapter mainly reviews the existing AMG systems. To introduce the general concept of music generation, I decompose the steps of music generation and elaborate on the main research topic in these steps. And then I review AMG systems by categorising their underlying generation methods.

Music generation can be factorised into several steps as shown in Figure 3.1.

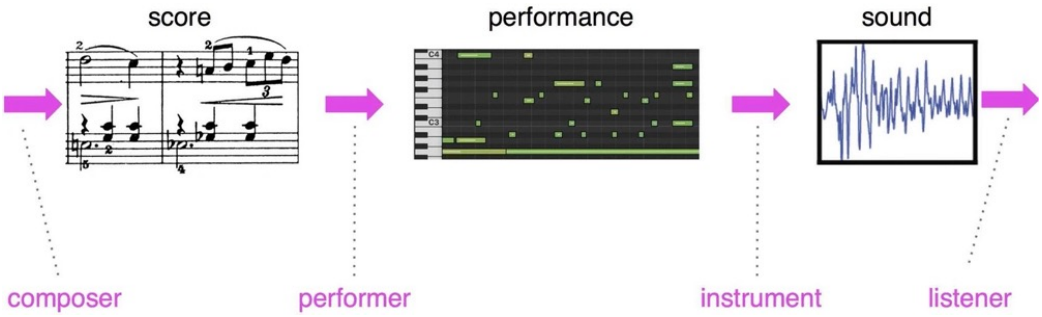


Figure 3.1: Steps of music generation (Oore et al. 2018).

The first step is to have the composer create the score. Such a process is simulated by many AMG systems in the symbolic domain (see Section 2.3 for symbolic representations). Although systems do not always generate from scratch, some works focus on harmonising the given melody. In the symbolic domain (see Section 2.3), many AMG systems have been trained to play the role of composer, which is mainly reviewed in this chapter.

The second step allows the score to be performed, the performer gives expressive timing and dynamics to the score by making subtle changes to start time, off time and velocity of notes. The same score can have different versions of performance, Aligned Scores and Performances (ASAP) dataset (Foscarin et al. 2020) contains musical scores in both MIDI and MusicXML formats and corresponding performances in MIDI and audio, models can be trained using such datasets to enable expressive rendering (Grindlay and Helmbold 2006; Flossmann et al. 2008; Chacón and Grachten 2016; Jeong et al. 2019a). Instead of separating the first and second steps, some AMG systems generate music with expressive timing and dynamics all at once (Oore et al. 2018; Huang et al. 2018). This is benefited from the serialisation method (see Section 2.3.2) allowing the encoded sequences to naturally have expressive timing and dynamics carried from the performances, so the models directly generate performances but cannot render expressiveness to scores.

The final step is to make the performance sound and perceived by listeners. This step is usually excluded in symbolic music generation, even though the choice of instruments and sound effects can dramatically change the actual listening experience.

Thus, when music is heard, it is evaluated from multiple perspectives, including compositional elements (e.g., rhythm, melody, harmony), performance and audio rendering. The comparative evaluation reported in Chapter 7 mainly focus on compositional elements.

With the continuous innovation of methodology in the field of music generation, each period is biased towards different approaches. The following sec-

tions aim to provide a chronological review of AMG systems in the symbolic domain. The review of systems is categorised into rule-based approaches, sequential models, artificial neural networks, and their successor, deep learning approaches.

3.1 Rule-based approaches

The relation between linguistics and music has long been discussed (Raffman 1993; Feld and Fox 1994; Patel 2003). Cooke (1959) presents an early work raising parallels between language and music, the focus is on investigating the common ground of ways that composers manipulate musical features to accomplish emotional expression. Through a series of profound analyses of hundreds of musical examples, the finding of the existence of conventional manipulation with specific meanings brings out the connection to language. Despite Dempster (1998) argues that musical structures are not homologous with language grammars as the necessity of encoding meanings, the concept of representing sequences of musical events or sound objects with formal grammars or automata (Hopcroft et al. 2001), which ignores meaning entirely, has been a majority tendency for rule-based approaches.

Many incipient experiments of (stored-program) computer-assisted composition mainly focused on new sound synthesis (Mathews et al. 1969; Doornbusch 2004), yet the string quartet –Illiac Suite– by Hiller Jr and Isaacson (1957) can be considered as the earliest findable examples of computer-written music work. It was obtained from a collection of experiments on various composition procedures, which were formalised into computational rules and probability functions to enable the generation of complex musical

textures. Around the same time, Xenakis (1992) initiates the application of mathematical models to music, including stochastic processing which has influenced research on algorithmic composition in the following decades.

Ebcioğlu (1990) introduces a constraint-based Bach chorales harmonisation system called CHORAL, which contains approximately 350 rules obtained by empirical observation and personal intuitions. Rules are represented by predicates (Boolean functions with arguments) in first-order logic and implemented by Backtracking Specification Language. For the tractability and convenience of maintenance, predicates are scheduled by a round-robin chain and categorised by various viewpoints of music (e.g., chord, melody, time), dependent views which share heuristics and constraints are further grouped into processes. The generation is then done by an iterative search for the next acceptable value based on history. However, as stated by Ebcioğlu (1990), using absolute rules to describe music is difficult, great composers always break them. In contrast, Bel and Kippen (1992) introduce Bol Processor. Unlike other constraint-based systems where rules are manually supplied, Bol Processor is a data-driven system utilising a grammatical inference algorithm to obtain rules. The early version demonstrates grammar-based generation with traditional drumming music. The drumming is transcribed to onomatopoeic labels and represented by strings (sequences of tokens), generative grammars are then learnt from the existing repertoire, which can be used to not only generate but also accept or reject a given string. Particularly, additional pattern rules in grammars were highlighted as an extension to ensure that periodic structures are revealed in the generated output.

Regarding more recent research, Quick and Hudak (2013) present probabilistic temporal graph grammars which are explicitly designed to address the problem of generated music not adhering to a certain structure such as repetition of phrases. Anders and Miranda (2010) aim to generalise constraint-based systems by utilising a higher-order function as constraint applicator which takes constraints as function arguments, so that the whole system is freely programmable. The replication of some previous constraint-based systems is then demonstrated by using the constraint applicator.

3.2 Sequential models

Musical dice games (Musikalisches Würfelspiel) of the eighteenth century (Hedges 1978) are an early example of probabilistic generation applied to Western music. The game begins with a set of prefabricated music components (e.g., notes in bars), from which a "new piece" is formed at random according to the outcome of the dice rolls. This stochastic process can be modelled by Markov models (Norris and Norris 1998), which were defined a century later. A first-order Markov chain (the simplest type of Markov model) consists of a finite state space, a transition matrix and an initial distribution. For example, one could encode pitch classes into states and assign a transition probability (or derive it empirically from music data) to each pair of states (Collins et al. 2011). The generation process begins with a starting pitch class sampled from the initial distribution, then repeatedly generating transitions between states to obtain a "new" sequence. Ames (1989) and Collins et al. (2011) provide overviews of the application of Markov models to AMG. Conklin and Witten (1995) introduce viewpoints as a means of building a

multi-dimensional Markov model, which is then optimised via prediction. Eigenfeldt and Pasquier (2010) propose a real-time system to generate harmonic progressions. This system acts as a composer assistant allowing users' input to influence the continuation selection instead of completely relying on machine selection. Allan and Williams (2005) applies hidden Markov models on chorale harmonisation, where the corresponding harmony is inferred with a given melody. Cope (1996, 2005) introduces Experiments in Musical Intelligence (EMI), which is a well-known program whose underlying generative mechanism appears to be that of a Markov model (e.g., Cope 2005), and which is said to have generated Bach chorales, Chopin mazurkas, and Mozart operas. The lack of full source code and description of how the model works have attracted criticism and called the EMI project into question (Wiggins 2008; Collins et al. 2016b).

Widmer (2016) states that the shortcoming of modelling music with history-based generation approaches, such as Markov models, will always be ineffective because any look-back, attention, or memory capability is inadequate with respect to music's long-term dependencies, which can span minutes and hours.[1] Collins (2011); Collins et al. (2016b); Collins and Laney (2017) have made several contributions that comprise nesting a Markov generator in another process that inherits the medium- and long-term repetitive structure from an existing, template piece, such that it is evident – on an abstract level – in the generated output (referred to hereafter as MAIA Markov). MAIA Markov is inspired by EMI, but unlike EMI, the source code has been made available[2]. Its outputs have been the subject of multiple, rigorously

[1]The deep learning for music literature tends to characterise "long term" as meaning evident temporal dependencies over the course of seconds; not minutes or hours.

[2]`https://www.npmjs.com/package/maia-markov`

conducted listening studies (Collins et al. 2016b; Collins and Laney 2017), and the starting point for use by artists in the AI Song Contest.[3]

Research by Gjerdingen (1988) on the Classical style suggests excerpts up to 4 bars in length can sound stylistically coherent without structural inheritance. When structural inheritance is required by a MAIA Markov user, it is accomplished by hard-coding a repetitive structure (e.g., reuse of bars 1–4 in bars 5–8) or running a pattern discovery algorithm such as SIARCT (Collins et al. 2013, 2010) to obtain one automatically. In the early version (Collins 2011; Collins et al. 2016b), the algorithm formalises each state as a pair consisting of (i) the beat of the bar on which a note, chord, or rest occurs, and (ii) the interval size between MIDI note numbers in that set, referred as a beat-spacing state. Subsequent work (Collins and Laney 2017) uses an alternative, beat-relative-MIDI state, due to superior performance: the state instead contains MIDI note numbers relative to an estimated tonal centre. MAIA Markov is one of the selected AMG systems that is further studied in Chapter 6 and 7.

3.3 Deep learning

Here I start by reviewing methods proposed during what has been referred to as the "AI winter" of the late 1980s and early 1990s. Todd (1989) describes the first application of neural networks to music generation, exploring various symbolic representations of music, and deciding on one-hot vectors for representing musical pitches. Todd demonstrates two different model architectures. The first is based on a feed-forward neural network, taking a fixed

[3] `https://aisongcontest.com`

time window of melodic information as input to predict the following window. The second comprises a sequential and recursive approach: similar to recurrent neural networks (RNNs) where the output is reentered into the input layer, the model input at each time step consists of a constant melody plan and memory of the melody so far.

The first proper RNN-based music generation model is CONCERT, introduced by Mozer (1994). Inspired by the psychological representation of musical pitch (Krumhansl 1979), the input representation named PHCCCH is formed by combining pitch information, the chroma circle, and the circle of fifths. Hild et al. (1991) introduces HARMONET, a system harmonising in the style of Bach chorales with given melodies. The system is hybrid, with RNNs generating notes and a second rule-based algorithm checking for "musical rule breaks" relative to the style, such as parallel fifths.

During the past decade, many deep learning generative models have been proposed recently for symbolic AMG (e.g., Sturm et al. 2015; Yang et al. 2017; Huang et al. 2018; De Boom et al. 2019; Tan and Herremans 2020). Like many sequential models, some deep learning models represent music as a sequence of tokens, where generation is carried out by repeatedly predicting the next token based on one or more previous tokens. Models based on standard RNNs often cannot effectively learn global musical structure. As an early attempt to address this problem, Eck and Schmidhuber (2002) use long short-term memory networks (LSTMs) to learn melody and chord sequences, and the generated blues music reveals stronger global coherence in timing and structure than using standard RNNs. In addition to different architectures, a point of comparison with approaches from the AI winter is that deep learning

researchers for AMG tend to favour minimal (if any) manipulation of the raw symbolic representation. For instance, whereas Mozer (1994) leverages contemporary, empirical research in music cognition that provides evidence for how music is perceived by, shapes, and subsequently interacts with human neural structures, Huang et al. (2018) and others rely on or assume the ability of neural network models to extract non-trivial (or cognitive-like) features without programming for them explicitly.

Oore et al. (2018) serialise polyphonic music and apply RNNs to generate output with expressive timing and dynamics. The serialisation converts notes into four-event sets: note-on, note-off, set-velocity and time-shift. But, as mentioned above, the output of RNN-based AMG models often lacks coherent long-term structure, much like simple Markov models. The multi-head self-attention mechanism of transformer models (Vaswani et al. 2017) shows promise in capturing long-term dependencies, and has been considered superior to RNNs in many tasks (Devlin et al. 2019). Huang et al. (2018) use the same serialisation method (Oore et al. 2018) to adapt a transformer model to generating music, calling it Music Transformer.[4] Benefiting from the self-attention mechanism, it achieves lower validation loss compared to the RNNs (Oore et al. 2018), and also longer-term stylistic consistency than previous RNN-based approaches. However, the underlying problem of insufficient originality of generation has not been paid attention to in the search field, it has been mainly discussed in the community. This problem will be

[4]Python dependency issues have caused multiple researchers including ourselves to be unable to reuse the published code at `https://github.com/magenta/magenta`. Using the published code, however, I was able to reimplement the same data preprocessing, the same model architecture with the same hyperparameters, and the same training procedure. Therefore, I continue to refer to this implementation throughout the paper as Music Transformer.

discussed in Chapter 6.

Thickstun et al. (2019) describe an architecture that learns directly from a music data representation called kern (no expressive timing and dynamics). The system combines both convolutional and recurrent layers for modelling horizontal and vertical note relations, and enables multi-instrument generation for music in various classical styles. Mao et al. (2018) explore deep learning models to generate music with diverse styles conditioned by a distributed representation. In addition to sequential generation, Hadjeres et al. (2017) use pseudo-Gibbs sampling to generate music in the style of Bach chorales. Yang et al. (2017) use the piano-roll representation to treat music as images, and train generative adversarial networks (GANs) with convolutional neural networks (CNNs) to generate music. Dong et al. (2018) apply the same overall method, but the generation is performed in a hierarchical manner, to capture the coherence between tracks and bars. Roberts et al. (2018) use variational autoencoders (VAEs) with LSTMs as both encoder and decoder; the results highlight that it can achieve better performance in capturing long-term dependencies than flat-RNNs baseline models. Here, Music Transformer (Huang et al. 2018), MusicVAE (Roberts et al. 2018) and the coupled recurrent models (Thickstun et al. 2019) are selected to study further in Chapter 6 and 7.

3.4 Discussion

This chapter reviews existing AMG systems, categorised by rule-based approaches, sequential models and deep learning. AMG can be achieved by various approaches, and not all existing work fits neatly into my categories.

For example, Herremans and Chew (2017) present MorpheuS, which regards music generation as an optimisation problem and applies a variable neighbourhood search algorithm to find solutions with the most appropriate notes.

Common criticisms of deep learning are the lack of interpretability of the learnt weights and the use of various training tricks that increase model performance in practice, and the need for large datasets to train, test, and validate the models, which does not reflect how a human composer can imitate the style of another (but not copy note sequences directly) based on just a few pieces of music. Rule-based systems have been considered expensive in design and not flexible in generating with a wide range of styles; but unlike with neural networks, one can easily trace the behavioural logic (generation decisions) of rule-based and sequential models. In contrast, according to some papers where deep learning has been used, these models can automatically and efficiently learn non-trivial features from complex data (Graves et al. 2013; He et al. 2016; Devlin et al. 2018).

When deep learning AMG models are evaluated, the comparison to other computational systems tends to focus on other deep learning models, overlooking existing *non*-deep learning approaches entirely (Huang et al. 2018; van den Oord et al. 2016; Roberts et al. 2018). I select several systems with different generating strategies to produce music excerpts for conducting the listening study (see Section 7.2.2), to fill the gap in comparative evaluation between deep learning and non-deep learning methods. In the next chapter, I review recent research on the evaluation methodology of creative systems, and highlight the evaluation conducted by deep learning-based AMG research work. And then I introduce Bayes factor analysis for hypothesis testing in

Chapter 7.

4 Literature review: Evaluation methodology

The necessity of evaluation is self-evident, as it determines whether the system achieve intending goals, and more importantly, it characterises systems, which could be largely beneficial to future research. There are multiple benefits to comparative evaluation, yet the amount of effort invested in evaluation methodology is less than that expended on the development of generative systems themselves (Pearce and Wiggins 2001; Jordanous 2012). While human-composed stylistic compositions have been evaluated for centuries (e.g., Cambridge University Faculty of Music 2010), there is a lack of explicit criteria or standardised methods according to which such evaluations are conducted. In this chapter, I first review evaluation frameworks for creative systems, and then I introduce two commonly used evaluation approaches for AMG systems: metrics-based objective evaluation and subjective listening study. The echo chamber review issue of deep learning-based AMG research is reported. Finally, I introduce Bayes factor analyses, which are used to verify the hypotheses with ratings collected in the listening study described in Chapter 7.

4.1 Evaluation of generative systems

O'Donoghue (2007) presents a generic approach to evaluating computational creativity by conducting statistical tests for analogies. Pearce and Wiggins (2001) provide a generic evaluation framework for AMG systems, which consists of four stages: identifying the goal of a system; defining a critic from examples in the target genre; generating music samples that satisfy the critic; evaluating the generated samples with human judges. Authors' opinions differ on the importance of evaluating the creativity or general details of the generation process itself (putting the outputs to one side): Pearce and Wiggins (2001), Boden (1990), and Cohen (1999) are in favour of such considerations, while Hofstadter (1995) argues that the internal mechanisms of a generation process can be inferred and appraised via repeated evaluation of its outputs. Pearce et al. (2002) formalise AMG into four areas or activities: algorithmic composition, the design of compositional tools, the computational modelling of musical styles, and the computational modelling of music cognition. In so doing, they attempt to make research aims more specific, and argue for applying evaluation methods appropriate to each area, thus addressing a prevalent failure in the field of not identifying clear motivations and goals for AMG. Inspired by the work of Pearce and Wiggins (2007), I take a similar approach of combining a listening study and hypothesis testing, to compare the performance of systems according to various musical dimensions. As such, this work falls into their third category: "computational modelling of musical styles".

Regarding a generic evaluation framework for creative system, Jordanous (2012) proposes SPECS (Standardised Procedure for Evaluating Creative

Systems), which consists of a three-stage process: 1) stating what it means for a particular system to be creative, 2) deriving tests based on these statements, and 3) performing the tests. The evaluation is supported by an empirical collection of key components of creativity. A use case of this methodology is presented for a music improvisation system. Although evaluation methods may vary individually, Jordanous (2019) also surveys four evaluation methods and summarises five criteria for evaluation methods themselves – in effect evaluating evaluation – which are correctness, usefulness, faithfulness as a model of creativity, usability of the methodology, and generality.

Agres et al. (2016) provide a wide-ranging overview of objective evaluation methodologies for computational creativity and their application to musical metacreation. These methods are categorised into external evaluation (e.g., Torrance 1998; Amabile 1982) and internal evaluation (e.g., Colton et al. 2014; Gardenfors 2004)). An evaluation is considered to be external when the source of judgments or measurements comes from outside of the system itself. Although the review highlights the importance of evaluating creative processes for a certain system, it is still unclear how to establish a standard across systems utilising different generation strategies. Agres et al. (2016) describe the advantage of questionnaire data for identifying which part of a system is not working as well as it might. However, a potential drawback is listeners' opinions can be altered by the very act of asking them (Schwarz 1999).

In terms of evaluation by metrics, Yang and Lerch (2020) introduce an evaluation framework for AMG models. Characteristics of a music dataset are determined by extracting musical features, which are then used further

to conduct kernel density estimation and obtain the Kullback–Leibler divergence between two datasets. In this way, one can investigate whether generated music demonstrates the same properties as the training data. Yang and Lerch (2020) utilise five pitch-based features: pitch count, pitch class histogram, pitch class transition matrix, pitch range and average pitch interval; four rhythm-based features: note count, average inter-onset-interval, note length histogram and note length transition matrix. A shortcoming of this approach is the simplicity of the features employed to date. For instance, it is possible to generate material that conforms with respect to basic pitch and rhythmic features, but still falls short of higher-level music-theoretic concepts that have been identified by musicologists (Rosen 1997; Gjerdingen 2007), and that may be perceived, implicitly or explicitly, by judges in listening studies.

The features-based evaluation approach (Yang and Lerch 2020) is relatively recent, whereas most deep learning AMG papers (e.g., Lim et al. 2017; Oore et al. 2018; Huang et al. 2018) evaluate generation performance and compare baselines using training/validation loss. For example, models that repeatedly predict the next tokens (e.g., musical events) are usually based on the concept of a language model in natural language processing. Negative log-likelihood/cross-entropy loss is used to help models find the statistically correct tokens, and certain advanced sampling processes can be used to adjust the randomness of generated sequences. But for generated music, loss value does not necessarily equate to strong performance along one musical dimension or another, only to the likelihood of being a valid sequence. The criterion used by variational autoencoder (Roberts et al. 2018) is a weighted

sum of cross-entropy and Kullback–Leibler divergence, where lower divergence weight increases the statistical accuracy of prediction but narrows the diversity of generation. Generative adversarial networks (Yang et al. 2017; Dong et al. 2018) optimise the generator and discriminator via a minimax tradeoff in which the discriminator evaluates generated output according to the probability of it being indistinguishable from training items. With respect to evaluation by human participants, some AMG papers do not include a formal listening study at all (Johnson 2017; Hadjeres and Nielsen 2020), while one invites composers to comment on the generated music (Oore et al. 2018). Other AMG papers do include listening studies (Liang 2006; Lim et al. 2017; Hadjeres et al. 2017; Mao et al. 2018; Roberts et al. 2018; Huang et al. 2018), but there is no standardised approach and sometimes no information is provided as to participant recruitment or musical backgrounds, and the constructs (e.g., "stylistic success") that are operationalised (turned into working definitions and written into questions) also vary widely. For instance, a paper on a harmonisation system may focus solely on whether "correct" chord identities are generated by the system, but omit a question regarding overall stylistic success.

4.2 Deep learning's echo chamber

While conducting the review of the AMG literature, I find many deep learning papers have "echo chamber" reviews and/or evaluations. I review nine papers on music generation with deep learning (Donahue et al. 2019; Oore et al. 2018; Roberts et al. 2018; Huang et al. 2018; Mao et al. 2018; Hadjeres et al. 2017; Yang et al. 2017; Dong et al. 2018; Thickstun et al. 2019),

and count the number of citations of non-deep learning music generation papers (research on music generation, harmonisation, expressive rendering, and style transfer are counted; survey papers are not). On average, 70% of a deep-learning-for-music paper's citations are to other deep-learning-for-music papers. At the extremum, only 27% citations of Hadjeres et al. (2017) are to other deep-learning-for-music papers, which implies a large coverage of other systems beyond deep learning, whereas 100% citations of Roberts et al. (2018) are to other deep-learning-for-music papers. It would appear that researchers of deep learning for music have focused on developing new architectures that are then subjected to metric-only evaluations in comparison to other deep learning methods, rather than on evaluating whether their system's outputs a) sound stylistically successful to listeners or b) compare favourably to outputs of non-deep learning systems. Researchers have been endlessly developing music generation systems of one kind or another. An analysis of concrete and actual system performance has been less concerned. This attitude of only asking for hard work and not about harvesting is certainly due to the rapid iteration of machine learning methodologies, but it is also closely related to the lack of awareness of the importance of evaluation.

4.3 Bayes factor analyses

A commonly observed characteristic of science, and one that is used to distinguish it from pseudoscience, is that scientific theories consist of falsifiable statements, which – in spite of attempts to falsify them via experiments – appear to stand the test of time (Goodwin and Goodwin 2016; Popper 2005, 2014).

Inferential statistics (e.g., t-tests, analysis of variance or ANOVA) is the branch of statistics via which conclusions may be inferred from observed data (Aron et al. 2013). Contrast this with descriptive statistics (e.g., mean, variance), which merely enable one to describe datasets. As such, inferential statistics are often used in pursuit of scientific progress, because the inferences made from data observed during experiments can be used to bolster or undermine particular theories.

The two main hypothesis testing frameworks in inferential statistics are called frequentist and Bayesian. A typical frequentist hypothesis test proceeds by stating a null hypothesis H_0 (e.g., no difference in performance between Systems A and B) and an alternative hypothesis H_1 (e.g., a difference in performance between Systems A and B), and determining a cutoff score on some comparison distribution such that the probability of observing this score in an experiment is less than a so-called, arbitrary p-value of .05. When the experiment has been conducted and a sample's score has been obtained, the experimenter either fails to reject H_0, or finds significant evidence for rejecting H_0 in favour of H_1.

A weakness of this approach is that one can never accept the null hypothesis, as the power of the test (stemming from the experimental design) could be insufficient to find a significant difference that may in truth be present.[1] Failure to reject H_0 is typically associated with a non-result, which is also regarded as a non-publishable result (Aron et al. 2013).

Returning to the example of the performance levels of Systems A and B, in a frequentist approach I can never infer that there is no difference in perfor-

[1] Power analyses are used to address this issue, but comprise assumptions that are themselves problematic.

mance; only fail to reject that there is no difference in performance. Suppose System A was a model for AMG with promising outputs, and System B was original, human-composed music in a target style. Using a frequentist approach, I could never be satisfied (or publish) that System A had attained the impressive level of System B, because I cannot infer no difference in performance.

A typical Bayesian hypothesis test also proceeds by stating H_0 and H_1, but, unlike a frequentist approach, *can* provide evidence in support of either null or alternative hypotheses. The fundamental shortcoming of the frequentist approach – of not being able to find in favour of the null hypothesis – is resolved in Bayesian hypothesis testing. Therefore, Bayesian hypothesis testing is better suited for experiments where systems are being compared to one another, and more generally to pursuing a scientific approach where theories consist of falsifiable statements (Dienes 2014; van Doorn et al. 2020).

The lack of a straightforward likelihood function has made the calculation of such tests difficult in the past, but Rouder et al. (2009) began to provide solutions for various common scenarios, at least where assumptions can be made about the normality of the underlying distributions (so-called parametric tests).

The counterpart of a p-value in the Bayesian approach is the Bayes factor BF_{10}, which is a likelihood ratio of the marginal likelihood of H_0 and H_1 (Dickey and Lientz 1970; Wagenmakers et al. 2010), given the equation

$$BF_{10} = \frac{Pr(\theta_0|H_0)}{Pr(\theta_0|data, H_1)} \tag{4.1}$$

where θ_0 denotes the parameter of interest. The value of BF_{10} can be inter-

preted as stated in Table 4.1 (Lee and Wagenmakers 2014).

Table 4.1: Bayes factor interpretation

BF_{10}	Interpreting
> 100	Extreme evidence for H_1
30 – 100	Very strong evidence for H_1
10 – 30	Strong evidence for H_1
3 – 10	Moderate evidence for H_1
1 – 3	Anecdotal evidence for H_1
1	No evidence
1 – 0.333	Anecdotal evidence for H_0
0.333 – 0.1	Moderate evidence for H_0
0.1 – 0.033	Strong evidence for H_0
0.033 – 0.01	Very strong evidence for H_0
< 0.01	Extreme evidence for H_0

Previous work tends to treat numeric data such as stylistic success ratings as though they are ratio-scale or at least equal-interval (Pearce and Wiggins 2007; Collins and Laney 2017), but in reality such assumptions are invalid. The data can be assumed only to be ordinal, and therefore a non-parametric Bayes factor test is required. van Doorn et al. (2020) introduce a way to calculate this test by data augmentation with Gibbs sampling, which lets the sample follow a latent normal distribution adapted to the non-parametric scenario. They also provide source code for demonstrating the Bayesian counterparts of three non-parametric frequentist tests: the rank-sum test, the signed rank test, and Spearman's ρ. I use the method of van Doorn et al. (2020), as it fits the circumstances here, with accessible code to conduct the following hypothesis tests.

To my best knowledge, this is the first use of such tests in evaluating the performance of machine learning AMG systems. Ideally, the introduction

to, and examples of using, the tests that I provide here will lead to wider uptake in the machine learning literature, because the Bayesian approach is preferable to the frequentist one, and van Doorn et al. (2020) offers tests appropriate for use with non-parametric data.

4.4 Summary

In this chapter, I first review the methodology for evaluating AMG systems (Pearce and Wiggins 2001, 2007), introduced the concept of internal (e.g., Colton et al. 2014; Gardenfors 2004) and external (e.g., Torrance 1998; Amabile 1982) evaluation. Metrics-based evaluation and listening study are two approaches commonly used to evaluate AMG systems. And recent deep learning-based AMG research tends to include listening study (Liang 2006; Lim et al. 2017; Hadjeres et al. 2017; Mao et al. 2018; Roberts et al. 2018; Huang et al. 2018). However, I also noticed the issue of "echo chamber" evaluations within the nine paper reviewed (Donahue et al. 2019; Oore et al. 2018; Roberts et al. 2018; Huang et al. 2018; Mao et al. 2018; Hadjeres et al. 2017; Yang et al. 2017; Dong et al. 2018; Thickstun et al. 2019). Finally, I introduce Bayes factor analyses, which is used for the hypotheses testing described in Chapter 7. And the proposed hypotheses are primarily aimed at comparing systems performance in the listening study. Since this chapter leads to the end of the background introduction and literature review, the next chapter states three major research questions to be addressed in this thesis.

5

Research questions

Related works and their issues in the field of AMG are discussed. The lack of comprehensive and standard evaluation methods leads to difficulties in understanding system performance in musical aspects and direct comparisons between systems. Here I develop three research questions around this main issue and describe how to address them in the following chapters, which lead to the main contribution of this thesis.

1. **To what extent the current models generate original music compared to human composers baseline?**

Music plagiarism issue has long been discussed (Müllensiefen and Pendzich 2009; Stav 2014; Neely 2019; Sturm et al. 2019b), but to my best knowledge, there is very little work addressing this issue by measuring originality, particularly in the field of machine learning. To answer the first question (see Chapter 6), the originality report is introduced for measuring the extent to which an algorithm copies the input music. Given that music has varying degrees of similarity depending on the style, the originality here is measured relative to a human baseline rather than an absolute level. To construct the originality baseline, the report is first applied to human-composed music to determine the extent to which human composers borrow from themselves

and each other. Two generation algorithms, MAIA Markov and Music Transformer, are trained with the same human-composed music, I then apply the report to the generated musical outputs and compare their originality to the human baseline. Furthermore, I substitute two similarity scores to investigate the originality of music with different characteristics.

2. **In a musicological aspect, what is the quality gap between:**

   (a) **deep and non-deep learning music generation systems?**

   (b) **human composers and these AI-based generation systems?**

As mentioned in Section 4.2, there are echo chamber issues in recent works of AMG with deep learning. Before this thesis, deep and non-deep learning generation systems have not been directly compared, especially in the musicological aspect. The answer to the second question (see Chapter 7) starts with defining six musical dimensions: stylistic success, aesthetic pleasure, repetition or self-reference, melody, harmony, and rhythm. I then prepare four generation systems including deep and non-deep learning methods. Nine hypotheses of systems' generation performance are listed, such as MAIA Markov and Music Transformer have no difference in stylistic success. To verify these hypotheses, I then conduct a listening study where 50 participants are recruited to 1) listen to the mix of human-composed and system-generated music excerpts; and 2) rate them in those six musical dimensions from one to seven. The results are finally interpreted by using non-parametric Bayesian hypothesis testing.

3. **What musical features can be extracted from music data to represent the quality of various musical dimensions?**

Knowing the ratings in musical dimensions helps to comparatively evaluate generation systems, but it does not necessarily reveal the particular features of music leading to the corresponding ratings. As conducting a comprehensive listening study is a relatively time-consuming process, having such features that are highly correlated with those musical dimensions creates a potential for automatic rating prediction and output evaluation. This leads to the answer to the third question (see Chapter 8), I introduce six types of features: statistical complexity, translational complexity, arc score, tonal ambiguity, time intervals and onset jitters. These features are then extracted from excerpts used in the listening study, and an exhaustive raincloud (Allen et al. 2019) report is provided that includes plots for all feature and rating pairs, I then discuss the effectiveness of each feature in representing musical dimensions based on this report.

6

Originality analysis

This chapter aims to address research question 1: "To what extent the current models generate original music compared to human composers baseline?" I first describe the background of music plagiarism, similarity and originality in AMG. I then introduce my method of measuring music similarity, and demonstrate the results of applying the originality report to MAIA Markov (Collins and Laney 2017) and Music Transformer (Huang et al. 2018), both are introduced in Chapter 3. The motivation for choosing Music Transformer mainly comes from its controversial generation performance, although it has proved to be quite effective in capturing long-term dependencies, the music it generates is also suspected of having a large copy from the training set (see Section 6.3). On the other hand, instead of focusing only on deep learning-based systems, I choose MAIA Markov as a representative of non-deep learning-based systems. MAIA Markov also shows excellent generation performance, especially the capability of having coherent musical structure, making it a suitable opponent for Music Transformer.

6.1 Music plagiarism

Music plagiarism is said to have occurred when there is demonstrable and perceivable similarity between two songs or pieces of music (hereafter, pieces), and when there is circumstantial evidence to indicate that the composer(s) of the latest piece would have been familiar with the existing piece. Stav (2014) describes how the musical dimensions of melody, harmony, and rhythm contribute to music plagiarism, and gives an example-based explanation of how these dimensions have been used in handling music copyright disputes. Based on the features of melodies involved in selected plagiarism cases, Müllensiefen and Pendzich (2009) derive an algorithm for predicting the associated court decision, and it identifies the correct outcome with 90% success rate. Recent failed or overturned cases also indicate that while music similarity and circumstantial evidence are necessary for delivering a verdict in favour of plagiarism having occurred, they are not sufficient, in that the distinctiveness of the music with respect to some larger corpus plays an important role too (Conklin 2010; Collins et al. 2016a; Neely 2019): melodies that share contours and begin and end on the same scale steps may well point to potential cases of plagiarism, but it is likely that other melodies will have these same characteristics too (Neely 2019); drum beats, where the initial space of possibilities is smaller compared to pitched material, have been less successful as bases for music plagiarism convictions (Otzen 2015).

Recently, discussions on ethical issues surrounding AI have attracted widespread attention. Collins et al. (2016b) use a note-counting approach to show that twenty bars of computer-generated musical output from an algorithm by Cope (Cope 2005) have 63% coincidence in pitch-rhythm com-

binations with a piece by Frédéric Chopin. According to Sturm et al. (2019b), a music generation algorithm's output and its tendency to copy original input pieces motivates the posing of open questions with respect to AI and music copyright law. As such generative models learn from existing music data, the copyright status of the output is unclear. Additionally, the evaluation of these models' outputs tends to be narrow; it does not involve any kind of *originality analysis* with respect to the human-composed pieces used for training. This creates copyright or plagiarism infringement risks for musicians who are using these algorithms as part of their creative workflows.

6.2 Approaches to music similarity

Largely outside of the role played by similarity in determining cases of music plagiarism, the systematic study of music similarity has a relatively long lineage (Selfridge-Field 1999) and continues to be of interest to scholars (Volk et al. 2016). One challenging aspect of studying the phenomenon is that two excerpts of music can be similar to one another in myriad ways (genre, instrumentation, timbre, tempo, dynamics, texture, form, lyrics, and mentioned above, melody, harmony, rhythm). This challenge interacts with variability in use cases too. Take a single paradigm such as query-based search in the form of music identification, which relies on some implementation of music similarity. Even for this one paradigm, there are various use cases: Shazam addresses the need for exact matching (Wang and Smith III 2012), a variant of SoundHound addresses query-by-humming (the user sings or hums at the interface and expects "successful" results),[1] and Folk Tune Finder allows

[1] `https://www.midomi.com/`

lyrics or notes to be input and, as with SoundHound's query-by-humming variant, the user's expectation of Folk Tune Finder is that the sought-after song will be found, or at least something relevant or interesting will be returned.[2] Of these use cases, only the one addressed by Shazam is clear cut – the other two are made more challenging by variation in cognitive and music-production capabilities of users, and there is not necessarily one "right answer".

Here, I am concerned with a more reductive view of music similarity – the type of note/fingerprint-counting approaches mentioned above. This is the characterisation of music similarity that a teacher might employ if a student's composition appears to draw too heavily on or copy directly from a known piece. For instance, "Why do 90% of the pitch-rhythm combinations in bars 1–20 of your piece occur also in this string quartet movement by Haydn?!" The representations and calculations required to reason this way, especially in an algorithmic fashion, began by Lewin (2007) and have been implemented in various forms since Ukkonen et al. (2003); Arzt et al. (2012); Collins et al. (2016a). In the next section, I define two similarity measures based on the P3 algorithm (Ukkonen et al. 2003) and the fingerprinting algorithm (Arzt et al. 2012).

The fingerprinting approach in music shares some commonalities with image matching (Lowe 2004) and perceptual hashing (Wang et al. 2015; Zauner 2010) in computer vision. Lowe (2004) aims to extract distinctive features from images and uses them to obtain the similarity to target images. This matching technique is shown to be robust to the affinity, noise, and change in viewpoint. Wang et al. (2015) develop a perceptual image hash method,

[2] `https://www.folktunefinder.com/`

which generates hash codes based on image features to address the problem of content authentication (or, similarly, copyright infringement in images and videos). Zauner (2010) proposes a benchmark framework for perceptual image hash functions, implemented in open-source software called pHash.[3]

6.3 Originality in music generation

As discussed in Section 3.3, a large number of deep learning models have been proposed for symbolic music generation (Roberts et al. 2018; Huang et al. 2018; Dong et al. 2018). Several of them regard music as a sequence of tokens, where generation involves predicting the next token based on previous tokens (Roberts et al. 2018; Huang et al. 2018). Oore et al. (2018) introduce a way to serialise polyphonic music and apply recurrent neural networks to generate output with expressive timing and dynamics. Huang et al. (2018) use this same serialisation to adapt a transformer model (Vaswani et al. 2017) to generate music. Benefiting from the self-attention mechanism, it achieved lower validation loss compared to the recurrent neural network of Oore et al. (2018) and also longer-term stylistic consistency than previous approaches which are based on recurrent neural networks. Based on the assumption that each musical output can be sampled from a normal distribution, Roberts et al. (2018) use variational autoencoders combined with long short-term memory networks. The application of generative adversarial networks and convolutional neural networks to music generation has also been explored (Dong et al. 2018), using the piano-roll representation and treating music as images that can be generated in a hierarchical manner.

[3] `https://www.phash.org/`

An issue with all the above deep learning approaches to music generation is that there has been inadequate consideration of music plagiarism in the algorithms' outputs. One user of the Music Transformer algorithm, Ruiz, writes:

> The thing is that I ran the code on my machine and it overfits. It needs a way to check that it isn't stealing from the dataset say no more than 6 or 8 continuous notes. If it can't do that it's useless. I mean your piano dataset is huge but after running the program for 20 times I found it composes note by note music of well known classical melodies. That's not OK. That should be avoided (Ruiz and Simon 2020).

Simon, a member of the Google Magenta team, replies:

> In the checkpoints we've released, we tried hard to reduce the ability of the model to perform pieces from the train set. And in the samples we released, we tried hard to remove any samples that are too similar to an existing piece of music. But it's difficult to get to 100% on these for a number of reasons, including the lack of a clear definition for "too similar" (Ruiz and Simon 2020).

Members of the general public can make use of Music Transformer for laudable reasons – Google Magenta have open-sourced the code – but their attempt to guard against music plagiarism appears problematic, and whatever constitutes "trying hard" in the above quotation has not been open-sourced, leaving general musicians who use Magenta algorithms in their creative workflows at risk of copyright infringement.

MAIA Markov (Collins and Laney 2017) is a non-deep learning approach to music generation that uses Markov models, pattern discovery, and pattern inheritance to ensure that generated material evidences long-term, hierarchical repetitive structure, also constitutes the first use of an **originality or creativity analysis** to assess the extent to which the model plagiarises human-composed works by J.S. Bach and Chopin on which it is based.

The remainder of this chapter studies two of the most promising models for music generation, Music Transformer (Huang et al. 2018) and MAIA Markov (Collins and Laney 2017), and focuses on the concept of originality, and methodologies for measuring it, which are then implemented and discussed.

6.4 Method

This section introduces the method I use to analyse the originality of one set of symbolically encoded music excerpts relative to another. I begin by defining the two sets of excerpts: the queries (the excerpts I am testing for originality), $\mathcal{Q}$, and the targets (the excerpts I am testing against), $\mathcal{R}$. Depending on the use case, $\mathcal{Q}$ may contain one or more excerpts from one or more pieces of music, and $\mathcal{R}$ usually contains overlapping excerpts from multiple pieces of music. The use cases for wanting to produce an **originality report**, which I explore further and exemplify, are as follows:

1. The user wants to determine the "baseline" level of originality within a corpus $\mathcal{C}$. In this instance, the queries $\mathcal{Q}$ are a (pseudo-)random sample from $\mathcal{C}$, drawn from pieces $\mathcal{Q}^*$, and the targets $\mathcal{R}$ are the set complement, $\mathcal{R} = \mathcal{C} \setminus \mathcal{Q}^*$. I distinguish between $\mathcal{Q}$ and $\mathcal{Q}^*$ because

if some excerpt $q \in \mathcal{Q}$ repeats or substantially recurs elsewhere in the piece Q^* from which it is drawn, and I leave this repetition in the set of targets $\mathcal{R}$, then q would be considered trivially unoriginal. Therefore, it is sensible to hold out entire pieces from which queries are selected. The outcome is a sample of N originality scores, from which estimates of the underlying distribution can be made, such as mean originality and confidence intervals about this mean.

2. The user wants to determine the level of originality of an algorithm's output relative to a corpus whose contents the algorithm is designed to imitate. In this instance, the queries $\mathcal{Q}$ would be overlapping excerpts of the algorithm's output, and I plot the originality of elements of $\mathcal{Q}$ as a function of time in the output, relative to elements of the target corpus $\mathcal{R}$. As well as plotting, I could also compare the mean or minimum originality found for the algorithm output to the distribution mentioned in the previous point, in which case the distribution acts as a "baseline" and can be used to address the question, "Is this algorithm's output sufficiently original?".

3. The user wants to incorporate originality reports into the modelling process itself, in order to analyse or steer/halt that process. The details are similar to the previous point, but the deployment of the method is during training or generation rather than after the fact.

The following sections explore the above use cases in turn.

6.4.1 Originality, similarity, and set of points

To implement the originality reports that are associated with each of the use cases, it is necessary to employ at least one similarity measure – that is, some function $c : \mathcal{Q} \times \mathcal{R} \to [0, 1]$, which takes two symbolically encoded music excerpts q and r and returns a value in the range $[0, 1]$, indicating q and r are relatively similar (value near one) or dissimilar (value near zero). The measure ought to be commutative, $c(q, r) = c(r, q)$, and have a identity-like property that $c(q, q) = 1$. The choice of similarity measure influences subsequent decisions with respect to addressing questions such as "Is this algorithm's output sufficiently original?" Here I define the ideas of use cases and the originality report, and illustrate them in the context of two particular similarity measures: the cardinality score and the fingerprinting score.

Each originality report centres on calculating an **originality score**, OS, for some query q in relation to the set of targets $\mathcal{R}$. In particular, I find the element $r \in \mathcal{R}$ that maximises the similarity measure $c(q, r)$, and subtract it from 1:

$$OS(q, \mathcal{R}) = 1 - \max\{c(q, r) \mid r \in \mathcal{R}\} \tag{6.1}$$

So the originality score $OS : \mathcal{Q} \times \mathbb{P}\,\mathcal{R} \to [0, 1]$, where $\mathbb{P}\,\mathcal{R}$ is the power set of $\mathcal{R}$, is also a measure in the range $[0, 1]$. A value near one indicates query q is original relative to set $\mathcal{R}$; a value near zero indicates it is unoriginal. If q is an exact copy of something that occurs in $\mathcal{R}$, then $OS(q, \mathcal{R}) = 0$.

6.4.2 Cardinality score

One particular similarity measure is the cardinality score, *cs* (Ukkonen et al. 2003; Collins et al. 2014; Janssen et al. 2019). Its strength is its simplicity and minimal parameter choices; its weakness is that if two sets of points (representing two note collections) differ from one another only by small but non-rigid amounts in one or more dimensions, *cs* will be near-zero, **contrary to perception** of the similarity between the two note collections still being high. The next section, on symbolic fingerprinting by geometric hashing, introduces an alternative similarity measure that addresses this weakness, but entails a larger number of parameter choices.

To calculate *cs*, I represent each music excerpt as a set of points containing the ontime[4] and numeric pitch representation (morphetic pitch number (Meredith 2006)) of each note.[5] So an element of the query set $\mathcal{Q} : \mathbb{P}(N \times N)$ is represented as

$$q = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \tag{6.2}$$

and an element of the target set $\mathcal{R} : \mathbb{P}(N \times N)$ is represented as

$$r = \{(x'_1, y'_1), (x'_2, y'_2), \ldots, (x'_{n'}, y'_{n'})\} \tag{6.3}$$

An example of this representation is provided in Figure 6.1(c) and (d). The bottom-left point in (c) has the value ($x_1 = 468, y_1 = 53$), representing an

[4]Ontime is the start time of a note counting in crotchet beats, with 0 for bar 1 beat 1 (Collins 2011).

[5]I use morphetic pitch in preference to MIDI note number here because the former is robust to major/minor alterations.

ontime at the beginning of the excerpt ($x_1 = 468$) and the morphetic pitch for C3 ($y_1 = 53$). The viola and cello have coincident notes at this moment, which project to a single point in the representation.

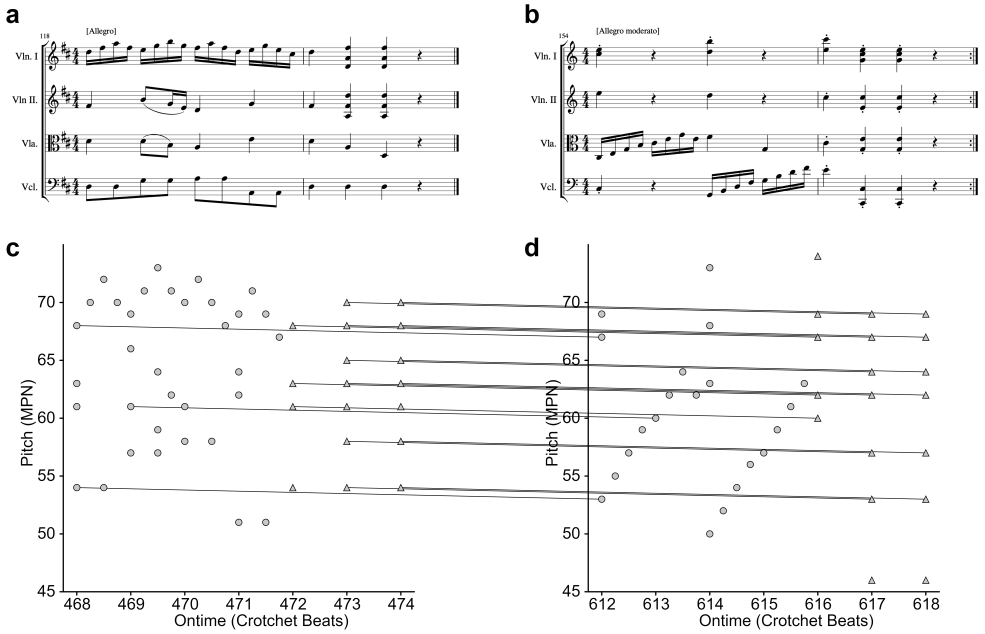


Figure 6.1: Visualisation of the cardinality score between two excerpts. (a) a 2-bar excerpt from Mozart; (b) a 2-bar excerpt from Haydn; (c) mapping notes in the excerpt (a) to a set of points; (d) mapping notes in the excerpt (b) to a set of points. For clarity, notes in the first/second bars are shown as circles/triangles.

Letting $\boldsymbol{t}$ be the translation vector that gives rise to the maximum cardinality of the intersection $(q + \boldsymbol{t}) \cap r$, I define the **cardinality score** as

$$cs(q, r) = |(q + \boldsymbol{t}) \cap r| / \max\{|q|, |r|\} \tag{6.4}$$

where $|q|$ is the size of the set of points q. I demonstrate calculations of the cardinality score with reference to the examples in Figure 6.1. In the top half of this figure, there are two excerpts of string quartets: (a) is by Mozart and (b) is by Haydn. Considering the set of points corresponding to bars 119

of the Mozart and 155 of the Haydn (second bars in both Figures 6.1(a) and (b), with corresponding points shown as triangles), the vector $\boldsymbol{t} = (-144, -2)$ translates 15 points from the Mozart excerpt to points in the Haydn, and the more numerous of the two sets is the Haydn excerpt, with 19 points, so the cardinality score is $cs(q, r) = 15/19 \approx 0.7895$. As a second example, considering larger point sets corresponding to bars 118-119 of the Mozart and 154-155 of the Haydn, the vector $\boldsymbol{t} = (-144, -2)$ translates 18 points from the Mozart excerpt to points in the Haydn, and the more numerous of the two sets is the Mozart excerpt, with 52 points, so the cardinality score is $cs(q, r) = 18/52 \approx 0.3462$.

6.4.3 Symbolic fingerprinting using geometric hashing

Another approach for similarity measuring is symbolic fingerprinting (Wang and Smith III 2012; Arzt et al. 2012; Collins et al. 2016a). Unlike the cardinality score above, where the similarity is measured note-wise allowing for one rigid transformation vector corresponding to time and/or pitch shifts, the symbolic fingerprinting approach generates hash entries from triples of notes that satisfy various constraints of pitch and time differences, and then measures the similarity via matching query entries to those in the lookup table, when both are encountered at the corresponding start time (refer to hereafter the start timing counting in seconds). The selection of triples, hash construction, and matching process each confer robustness to non-rigid differences between two note collections, as I describe and demonstrate below, and which helps increase the perceptual validity of symbolic fingerprinting using geometric hashing as a similarity measure compared to cardinality score.

For geometric hashing, I again represent music excerpts as sets of points (see Section 6.4.2), and store the ratio (as similarity) of the number of matching hashes across a dataset of pieces to the number of hashes generated by a query.[6] Entries in the lookup table are constructed from target excerpts in advance. I first prepare each excerpt as a set of (start time, pitch) pairs (sorted by ascending start time), and then I generate hash entries of valid triples searched via Algorithm 1. A valid triple contains three pairs of start times and pitches that satisfy time/pitch-difference constraints. For example, the triple V is denoted as:

$$V = \{v_0, v_1, v_2\} \tag{6.5}$$

where each v has two attributes *start* and *pitch*. Regarding the pitch dimension, I obtain the pitch difference between two adjacent points, denoted as pd_0 and pd_1:

$$pd_0 = v_1.pitch - v_0.pitch \tag{6.6}$$

$$pd_1 = v_2.pitch - v_1.pitch \tag{6.7}$$

I also calculate the intervals between start times, and obtain the ratio *tdr* instead of the time difference, as *tdr* supports tempo-independent search (in other words, as per Lewin (2007), the effect of scaling up/down the start

[6]The implementation of fingerprinting algorithm is contributed by Collins and Coulon (2019).

time values can be mitigated or factored out entirely):

$$td_0 = v_1.start - v_0.start \tag{6.8}$$

$$td_1 = v_2.start - v_1.start \tag{6.9}$$

$$tdr = \begin{cases} td_0/td_1 \; if \; td_0 \geq td_1 \\ td_1/td_0 \; if \; td_1 > td_0 \end{cases} \tag{6.10}$$

I define a triple being valid as its pd_0, pd_1 and tdr being in a certain range. I then encode pd_0, pd_1 and tdr into a string as the hash entry, with the following steps:

1. For each of pd_0 and pd_1, insert a + or − sign indicating a positive and negative relationship, and the absolute value of the pitch difference (e.g., +02, −10);

2. And then insert + if $td_0 \geq td_1$ and − if $td_1 > td_0$, preceding the float number of the tdr rounded to one decimal place.

For example, the entry string shown at the first item in Figure 6.2(c) is +7 + 10 + 4.0, because the interval from D3 to D4 is +7 MPNs, and the interval from D4 to G5 is +10 MPNs; the time difference between the D3 and D4 is 1 crotchet beat, and the time difference between the D4 and G5 is 0.25 crotchet beats, which yields $tdr = +4.0$ as the time difference ratio.

Algorithm 1 consists of nested loops aiming to find valid triples, which are then encoded into an entry and inserted into a lookup table. The outer loop iterates through each point denoted as v_0. The nearest inner loop iterates through the points after v_0, during which I apply the following constraints to find the second point v_1: pitch difference between v_0 and v_1, pd_0, is in the

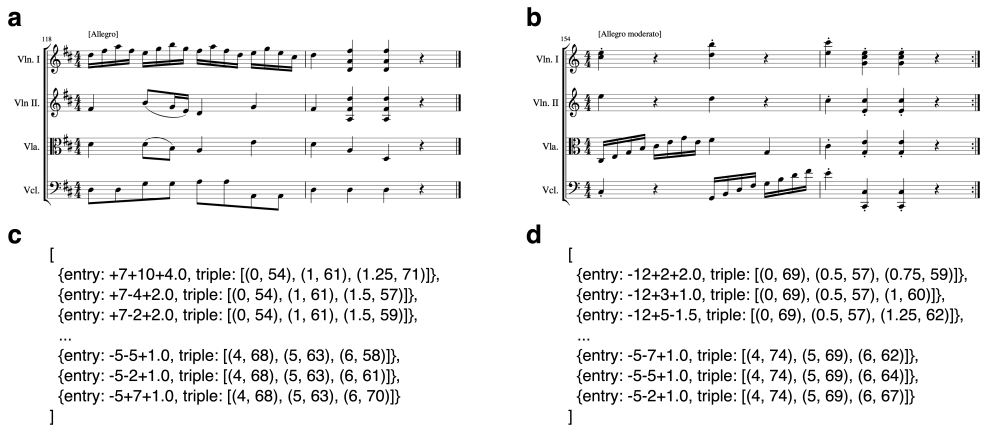


Figure 6.2: Entries generated for two excerpts. (a) a 2-bar excerpt from Mozart; (b) a 2-bar excerpt from Haydn; (c)&(d) a list of entries generated following the corresponding triples of (start time, pitch)-pairs in the excerpt (a)&(b).

range between $pMin$ (minimum pitch difference, e.g., 1) and $pMax$ (maximum pitch difference, e.g., 6); the start time difference between v_0 and v_1, td_0 is in the range between $tMin$ (minimum time difference, e.g., 0.5) and $tMax$ (maximum time difference, e.g., 2). If the conditions fail, the loop skips to the next iteration. The most inner loop further iterates through points after v_1, and then applies the same constraints to find v_2 to obtain pd_1 and td_1.

With a query q, I take the point-set representation and apply the same method as above to obtain entries. Each query entry is then used to match with the target entries saved in the lookup table, returning a point set $\{(a_1, b_1), (a_2, b_2), \ldots, (a_M, b_M)\}$ of (target start time, query start time)-pairs. In a scatter plot of such data, true positives appear as (approximately) diagonal lines. Thus, I apply a simple transformation (e.g., $(a_i, b_i) \rightarrow (a_i, a_i - b_i)$) and calculate a histogram over this transformed data. Supposing there are k unique entries generated from a query, I calculate the similarity ratio to a certain target by dividing the number of unique matches h in a histogram

Algorithm 1 Create entries from points

```
1: pts ← the set of points
2: npts ← pts.length
3: tMin ← 0.5; tMax ← 2; pMin ← 1; pMax ← 6
4: lookup ← a hash table which maps entries to arrays of start times
5: for (i = 0; i < npts − 2; i++) do
6:     v0 ← pts[i]
7:     j ← i + 1
8:     while (j < npts) do
9:         v1 ← pts[j]
10:        pd0 ← abs(v1.pitch − v0.pitch)
11:        td0 ← v1.start − v0.start
12:        if ((td0 > tMin) ∧ (td0 < tMax) ∧ (pd0 ≥ pMin) ∧ (pd0 ≤ pMax))
   then
13:            k ← j + 1
14:            while (k < npts − 1) do
15:                v2 ← pts[k]
16:                pd1 ← abs(v2.pitch − v1.pitch)
17:                td1 ← v2.start − v1.start
18:                if ((td1 > tMin) ∧ (td1 < tMax) ∧ (pd1 ≥ pMin) ∧ (pd1 ≤
   pMax)) then
19:                    entry ← create_entry(v0, v1, v2)
20:                    lookup.insert(entry, v0.start)
21:                end if
22:                if (td1 ≥ tMax) then
23:                    k ← npts − 1
24:                end if
25:                k++
26:            end while
27:        end if
28:        if (td0 ≥ tMax) then
29:            j ← npts − 1
30:        end if
31:        j++
32:    end while
33: end for
```

bin by k.

6.5 Originality reports

The datasets I use in this experiment are:

1. 71 MIDI encodings of Classical string quartet scores (sheet music) from the website KernScores.[7] This dataset is prepared according to the following filters and constraints:

    - string quartet composed by Haydn, Mozart, or Beethoven;
    - first movement;
    - fast tempo, e.g., one of Moderato, Allegretto, Allegro, Vivace, or Presto.

2. 1,276 MIDI encodings of performances on Yamaha Disklaviers by professional pianists of classical works from J.S. Bach to Alban Berg, from the MAESTRO dataset version 3 (Hawthorne et al. 2019).[8] This dataset is as follows:

    - the annotated train set consists of 962 performances, the annotated validation set of 137 performances, and the test set of 177 performances;
    - if there is a performance of piece A in the training set, no other performances of piece A appear in the validation or test sets. Vice

[7] See `https://osf.io/96emr/` for the datasets, algorithms, and analyses.

[8] The difference between Classical and classical is meaningful. The large-"c" "Classical" period refers to Western art music composed in the period 1750–1830, whereas the small-"c" "classical" period subsumes this, comprising the period between 1650–1920.

versa, if there is a performance of piece B in the validation or test set, no other performance of piece B appears in the train set.

The difference in the provenance of these two datasets (scores and performances) is important: a music score contains note start times and durations only up to some small integer subdivision of the main beat; a music performance captured via the MIDI format contains start times and durations in seconds (i.e. greater timing granularity). There are also differences in dynamic level (loudness) and indications of which notes should be played by which instrument or hand, but they are not relevant to the current work, so I will not elaborate on these further. In other words, the MAESTRO dataset contains the trace of human expressivity present in a performance, whereas the KernScores dataset does not. It is not necessarily the case that having the expressive note start and duration timing data is advantageous, compared to the reduced set of values available in a score representation, however. Ideally, one would have access to both types of timing data, **and the mapping between the two**. The MAESTRO dataset does not contain such a map, but I can still make use of it in this work to address two questions:

1. How does the originality of output from the Music Transformer algorithm change as a function of dataset, moving from a smaller dataset to the larger one on which its original publication is based (Huang et al. 2018)?

2. Does the concept of the originality report stand up to scrutiny when I switch from a straightforward similarity measure that assumes score-like note start times (cardinality score, Section 6.4.2), to a more com-

plex similarity measure that works also with expressive note start times (geometric hashing, Section 6.4.3)?

Sections 6.5.1-6.5.3 focus on originality reports obtained using cardinality score, and compare the performance of two algorithms in terms of the originality of their outputs. Then Section 6.5.4 provides analyses that address the two questions stated above.

6.5.1 Determining the originality baseline

To form the query and target sets for Classical string quartets dataset, I divide the 71 excerpts into two sets: 50 queries $\mathcal{Q}$ are drawn from 7 pieces $\mathcal{Q}^*$, and the targets $\mathcal{R}$ consisted of the remaining 64 pieces. The selection of the 7 pieces is pseudo-random to reflect the representation of composers and time signatures in the overall dataset. I use a fixed window size of 16 beats for each query.

To demonstrate the robustness of geometric hashing to music with expressive timing, the same approach is applied to the MAESTRO dataset (Hawthorne et al. 2019), using the 962 performances in the train set as the targets $\mathcal{R}$, and 50 queries $\mathcal{Q}$ drawn from 314 performances $\mathcal{Q}^*$ consisting of the amalgamation of the 137 validation and 177 test performances. I use a fixed window size of 8 seconds for each query.

I run the code outlined in Algorithm 2 to obtain the baseline for both approaches by using the corresponding similarity function c.

For the sample of $N := 50$ excerpts from Haydn, Mozart, and Beethoven string quartets (KernScores), the mean originality is $\text{mean}(OS) = 0.699$, with bootstrap 95%-confidence interval 0.672 and 0.725. I interpret this to mean

Algorithm 2 Estimating the self-originality of a corpus

Require: $\mathcal{Q}$, $\mathcal{R}$, query and target corpus, respectively.
Initialize O to an empty output list.
Initialize N to the number of originality scores.
for $(i \coloneqq 0, i < N, i++)$ **do**
    $q \coloneqq \text{sample}(\mathcal{Q})$
    Initialise C as an empty list to store similarity scores.
    for each $q \in \mathcal{Q}$ **do**
        $C.append(c(q, \mathcal{R}))$
    end for
    $O.append([1 - \max(C)])$
end for

that for the current corpus and sample (space of fast, first-movement Classical string quartets), composers wrote music that is 69.9% original, at least according to the note-counting music similarity measure employed here. For the sample of $N \coloneqq 50$ excerpts from classical piano performances (MAESTRO), the mean originality is $\text{mean}(OS) = 0.562$, with bootstrap 95%-confidence interval 0.546 and 0.579. As both the dataset and similarity metric have changed, it is not meaningful to compare the two confidence intervals to one another. But when music-generation algorithms are trained on one or the other corpus, it is meaningful to compare the originality of their outputs to their respective training corpus' self-originality, addressing the question of whether the music generation algorithm's output is sufficiently original.

With the parameters $tMin \leftarrow 0.5$, $tMax \leftarrow 2$, $pMin \leftarrow 1$, $pMax \leftarrow 6$, the lookup table has a maximum $8,928$ unique hashes. This is because there are 31 possible tdr's, $1.0, 1.1, \ldots, 4.0$, as well as 2 possible signs for each (see Equation 6.8). There are 2 pitch differences encoded in a hash (see Equation 6.6), covering 6 possibilities $1, 2, \ldots, 6$, again with 2 possible signs for each. This gives $8,928 = (31 \cdot 2) \cdot (6 \cdot 2)^2$ unique hashes in total. During the

building of the lookup table across the MAESTRO train set, each of these possible hash codes is encountered, and I list the ten top and bottom ten entries ordered descending by frequency of occurrence in Figure 6.3. From a musical point of view, the pitch content of the ten most common hashes is not surprising: they articulate rising and falling (or falling and rising) perfect fourths, (±5 difference in MNNs), major seconds (±2 difference in MNNs), and minor thirds (±3 difference in MNNs). It has also been found previously that time difference ratios of close to 1 (e.g., ±1.1) are more common than time difference ratios of exactly 1, suggesting that it is more desirable for pianists not to play such sequences absolutely isochronously, but to play one of the two-time differences slightly longer than the other (Widmer 2003). As for the least common hashes, it is not surprising to find extremely imbalanced time-difference pairs (±4.0) are rare, and include intervals of the tritone (±6 difference in MNNs) and minor second (±1 difference in MNNs).

Top 10

```
[
 { hash: +05-05+1.1, count: 268027 },
 { hash: +05-05-1.1, count: 264901 },
 { hash: -05+05+1.1, count: 255960 },
 { hash: -05+05-1.1, count: 255934 },
 { hash: +02-02+1.1, count: 254712 },
 { hash: +02-02-1.1, count: 253723 },
 { hash: -02+02+1.1, count: 248714 },
 { hash: -02+02-1.1, count: 246992 },
 { hash: -03+03+1.1, count: 237036 },
 { hash: -03+03-1.1, count: 235844 }
]
```

Last 10

```
[
 { hash: -02+01-4.0, count: 21 },
 { hash: -02+01+4.0, count: 21 },
 { hash: -01+02-4.0, count: 21 },
 { hash: +04-01-4.0, count: 21 },
 { hash: -06+04+4.0, count: 21 },
 { hash: -06-04+4.0, count: 21 },
 { hash: -02+06+4.0, count: 21 },
 { hash: -01-01-4.0, count: 20 },
 { hash: -04-06+4.0, count: 19 },
 { hash: +01+05-4.0, count: 19 }
]
```

Figure 6.3: The top 10 and last 10 hash entries that are generated as lookup table keys, with the descending order of occurrence count.

6.5.2 Is this algorithm's output sufficiently original?

The mean and confidence interval calculated above can be used to help address the question of whether an algorithm's output is sufficiently original. Assuming there is a passage generated by an algorithm, and I traverse that output, collecting n-beat excerpts with 50% overlap, say, into a query set $\mathcal{Q}$. In the experiment with cardinality score, I use $n := 8, 16$ beats, which corresponds to 2- and 4-bar excerpts in 4-4 time, respectively. It is advisable to use at least two different window sizes, to probe the assumption that originality should increase with window size. In other words, different window sizes can be used to determine whether a worrisome-looking instance of low originality at the 2-bar level increases – and so becomes less worrisome – at a longer 4-bar window size.

I run the MAIA Markov (Collins and Laney 2017) and Music Transformer algorithms (Huang et al. 2018) to explore this question of sufficient originality, based on the training data of 64 string quartet movements described above. The Music Transformer model's training data is augmented by transposing the original pieces in the range $[-5, 6]$ MIDI notes, and then I slice these into subsequences of fixed size 2,048 for batch training, giving a train set of 4,128 and a validation dataset of 564 subsequences. The model, with six layers, eight heads, and hidden size of 512, is trained with smoothed cross-entropy loss (Müller et al. 2019) and the Adam optimiser (Kingma and Ba 2014) with custom learning rate schedule (Bengio 2012). In keeping with the standard approach, the training process is stopped at epoch (checkpoint) 3, where the validation loss reached a minimum value of 1.183. Afterwards, 30 excerpts are generated by MAIA Markov and Music Transformer to form the query

set $\mathcal{Q}$, based on which the mean value of originality scores are obtained by following the same method in Section 6.4.1, now for each time window as the excerpt is traversed.

Figure 6.4 aims to provide insight into how the originality (cardinality) scores are distributed for both human-composed excerpts and model-generated excerpts, I obtain originality scores from 7 query pieces (human-composed) and 30 pieces generated by the model at epoch 3. It shows that the originality scores of human-composed excerpts have higher kurtosis, which implies that the model is less likely to generate excerpts within a narrow range of originality. The mean values denoted by the dashed lines also indicate the overall originality of generated excerpts is lower than the human-composed excerpts.

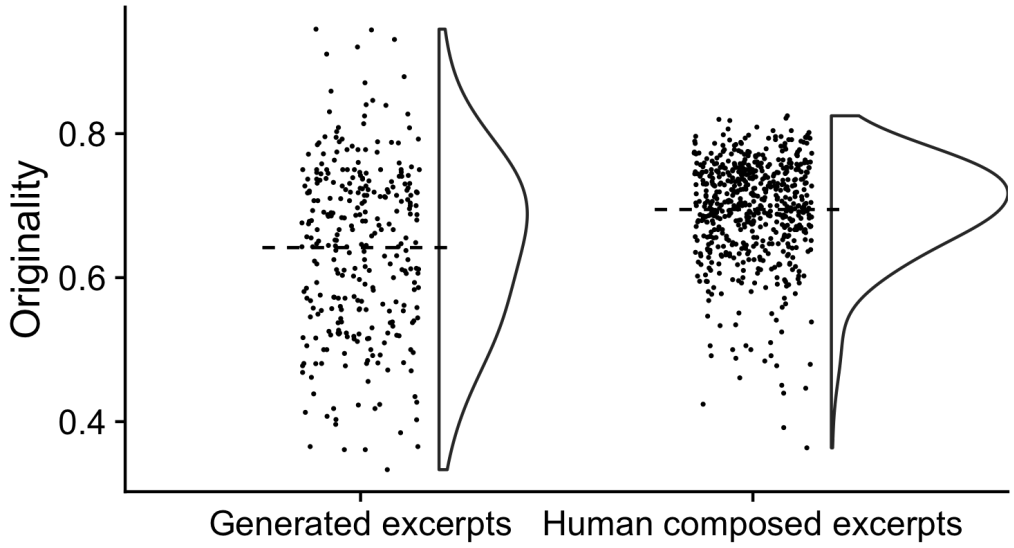


Figure 6.4: A rain cloud plot of originality scores for both model-generated excerpts and human-composed excerpts, with the dashed lines showing mean values.

For both algorithms, I see in Figures 6.5(a) and (b) that the originality at the 2-bar level is low relative to the mean and 95%-confidence interval for the baseline, but this is to be expected because the baseline is calculated at the 4-bar level. What I expect to see is the solid line – indicating algorithm originality at the 4-bar level – lie entirely inside that confidence interval. This is the case for MAIA Markov (Collins and Laney 2017), but Music Transformer's (Huang et al. 2018) mean originality level is entirely below this confidence interval, indicating it has issues with borrowing too heavily from the input on which it is trained.

Three typical worst case examples of copying are shown in Figures 6.5(c), (d), and (e), with generated outputs on the left and original excerpts on the right. Figure 6.5(c) shows one from MAIA Markov having 42.9% originality associated with Beethoven's String quartet no.6 in B-flat major, op.18, mvt.1, bars 61-64. And then, Figure 6.5(d) shows one generated by the "best" checkpoint (checkpoint 3 with the minimum validation loss) of Music Transformer having 48.8% originality associated with Mozart's String quartet no.13 in D minor, K.173, mvt.1, bars 125-128. I find most of the generated outputs in this stage with less than 50% originality are due to repeating the same note, which is also frequently found in Classical string quartets, and the model tends to start by reproducing this simple kind of pattern. Finally, Figure 6.5(e) shows one generated by checkpoint 15 of Music Transformer having 9.9% originality associated with Beethoven's String quartet no. 1 in F major, op.18, mvt.1, bars 233-234. Generally, the model appears to overfit at this checkpoint. I infer from these originality reports and basic musicological interpretations that the results generated by Music Transformer gradually

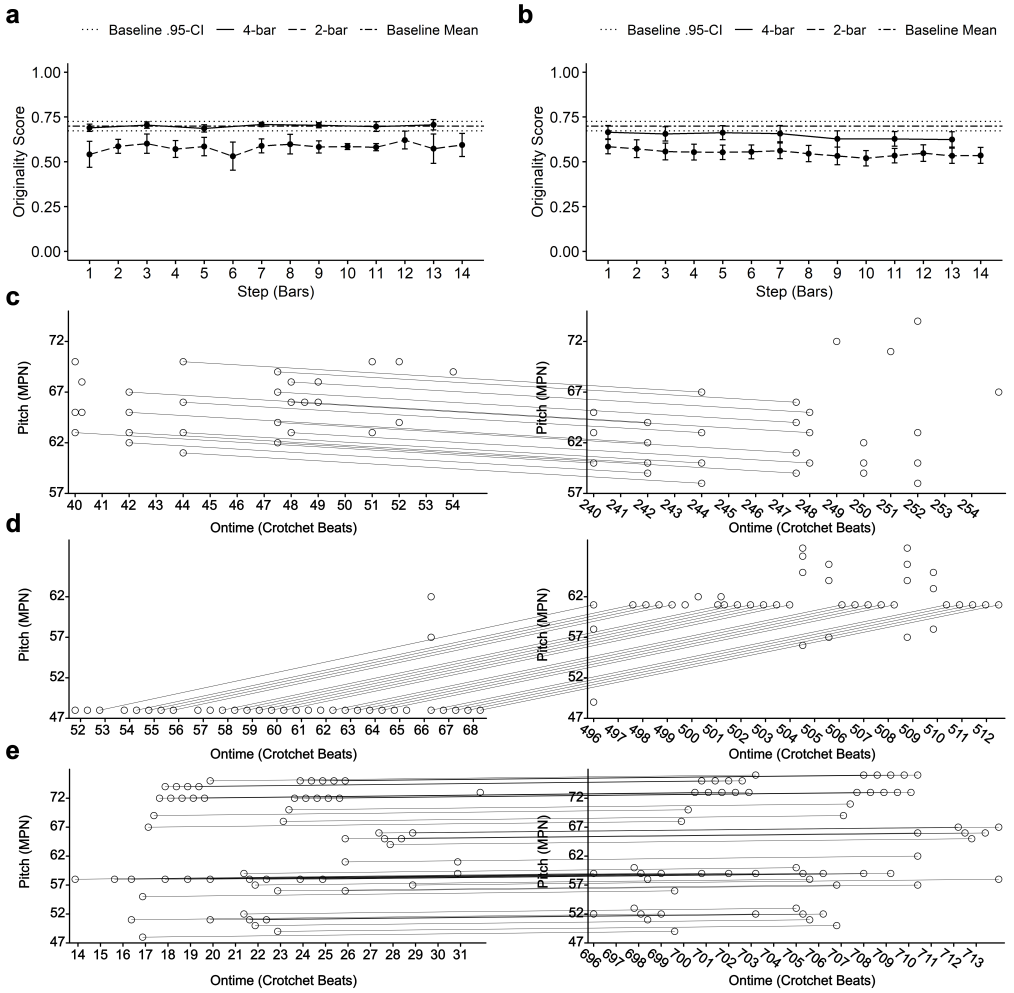


Figure 6.5: Originality report for the MAIA Markov and Music Transformer algorithms. (a) and (b) show the change in originality scores over the course of the excerpts obtained for MAIA Markov and Music Transformer, respectively, at 2- and 4-bar levels compared to the baseline mean and 95%-confidence interval; (c), (d) and (e) show worst-case examples of copying by MAIA Markov and Music Transformer at checkpoints 3 and 15, respectively, where the generated outputs are on the left and the human-composed excerpts are on the right.

morph during training from reproduction of simple patterns (e.g., repeated notes) to verbatim use of more distinctive note sequences.

6.5.3 Originality decreases as epoch increases

Here, I demonstrate the use of an originality report in the modelling process itself, as a means of analysing changes in originality as a function of model training or validation epoch. Music Transformer is used as an example of a deep learning model, with the train/validation set as in Section 6.5.1. To monitor the originality change along with the training process, 10 checkpoints including the initial point are saved. Again, I use each of them to generate 30 excerpts, to which the aforementioned originality report is applied. Afterwards, I calculate the mean value of those 30 originality scores for each checkpoint.

Figure 6.6(a) and (b) show the change of loss and accuracy respectively over training. As mentioned previously, the standard training process would stop at epoch (checkpoint) 3, where the validation loss reaches a minimum, but I extend the training process further to more fully investigate the effect of training on originality. Figure 6.6(c) and (d) contain a dashed line indicating the baseline originality level of 0.699 for the string quartet dataset. It is worth noting that Figure 6.6(c) and (d) start with epoch 1, instead of epoch 0 where the weights of networks are randomly initialised. Such difference is due to the near impossibility of forming notes from randomly generated sequences, because the serialisation method used by Music Transformer requires a matched pair of events to construct a note (Huang et al. 2018). Thus, applying the originality report for epoch 0 is unconsidered here. In

Figure 6.6(c), the mean originality score decreases as a function of the model training epoch, but remains largely in the 95%-confidence interval of the baseline originality level of the corpus. Figure 6.6(d) is more concerning, indicating that the minimum originality score decreases to well below the 95%-confidence interval of the baseline originality level of the corpus. Originality decreases until epoch 3, and then it stays relatively flat afterwards. However, as with the discussion of Figures 6.5(c) and (d) in the previous subsection, I find that the model's borrowing still becomes more verbatim (or distinctive) after epoch 3, thus originality in a more general sense is still decreasing, a fact that is not immediately evident from Figure 6.5 because the cardinality score does not consider distinctiveness, discussed further below.

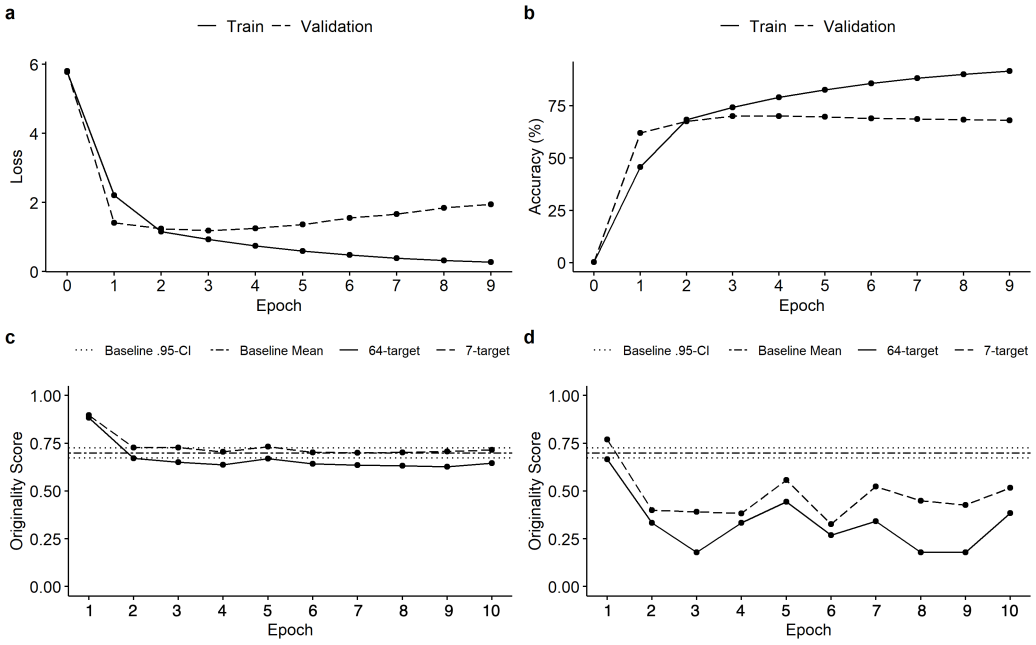


Figure 6.6: (a) the loss curve of train and validation; (b) the accuracy curve of train and validation; (c) the mean originality curve for 64-target and 7-target sets; (d) the minimum originality score curve for 64-target and 7-target sets.

6.5.4 Measuring with geometric hashing and exploring associated parameters

In this section, I demonstrate the originality report with a similarity measure based on geometric hashing, and its adaptability to music with expressive timing.

This time, I use the trained version of Music Transformer from the original publication (Huang et al. 2018) (which used the MAESTRO train set), and generate 30 passages of "new" music. I then traverse each of these passages and collect 8-second excerpts with 4-second overlap. To adapt with expressive timing, the time difference calculated in this case is in seconds instead of crotchet beats. As shown in Figure 6.7, Music Transformer's (Huang et al. 2018) mean originality level is again entirely below the self-originality confidence interval of the corpus, again indicating it has issues with borrowing too heavily from the input on which it is trained.

The adaptability of the geometric hashing approach to data with expressive timing is already implicit in the results shown in Figure 6.7, but I need to verify that: (1) when I define a query based on an excerpt of music that is drawn from (known to be in) the train set itself, I get a near-1 similarity score when the timing data in the query is perturbed by or subject to minimal jitter; (2) there is some reduction in similarity score as a function of jitter being increased from barely noticeable (around 1 msec) to barely recognisable as the same query (around 100 msec).

For each (time, pitch)-pair in a randomly selected query, I add random, uniform jitter with range $\pm x$ sec to the start time. In this case, I sample queries from the train set itself, and match them to the lookup table calcu-

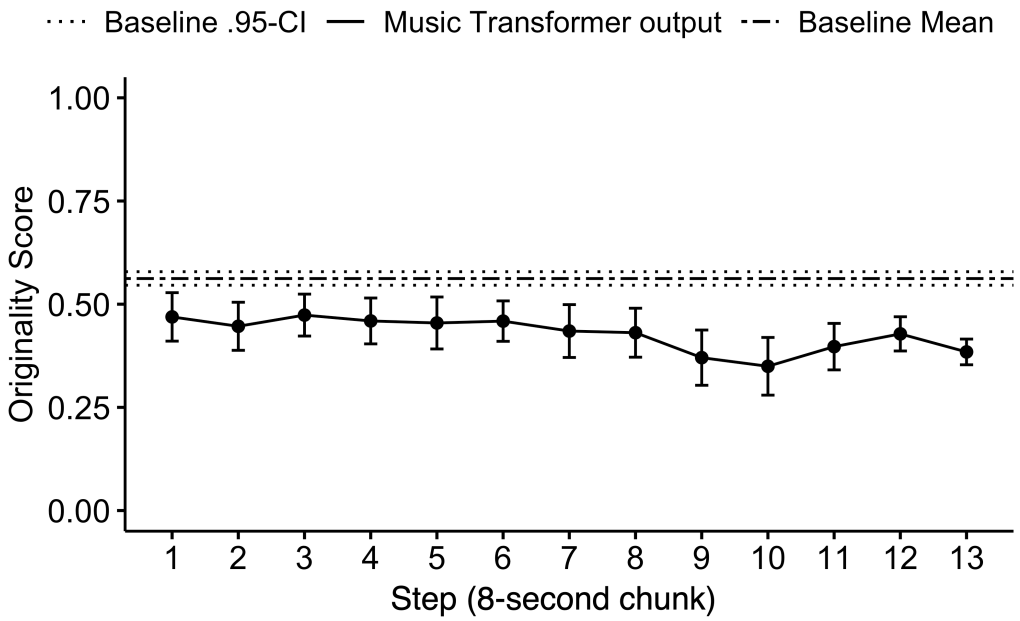


Figure 6.7: Originality report based on geometric hashing for the Music Transformer algorithm, which shows the change in originality scores over the course of the excerpts obtained for Music Transformer at 8-second levels compared to the baseline mean and 95%-confidence interval.

lated previously without jitter. I also verified that for no jitter and a bin size of 0.1 sec (fixed for this analysis), each query is able to produce maximal similarity 1 to the same piece from which is sampled.

As shown in Figure 6.8, the similarity values decrease as the jitter increases, which is what I expected. It is worth noting that there is a slight increase in similarity score moving from 0.05 sec jitter to 0.1 sec. This is because when I obtain the top m similarity scores and their associated pieces, I use $m = 100$ and look for the similarity score associated with the piece from which the query is drawn. But as jitter increases to 0.05 sec and beyond, I notice that often this piece is not among the top 100 matches. In such cases, I report the maximal similarity that is found (to some other piece), and hence the right-hand two entries in this plot may be slightly inflated, compared to one another and/or to the left-hand three entries.

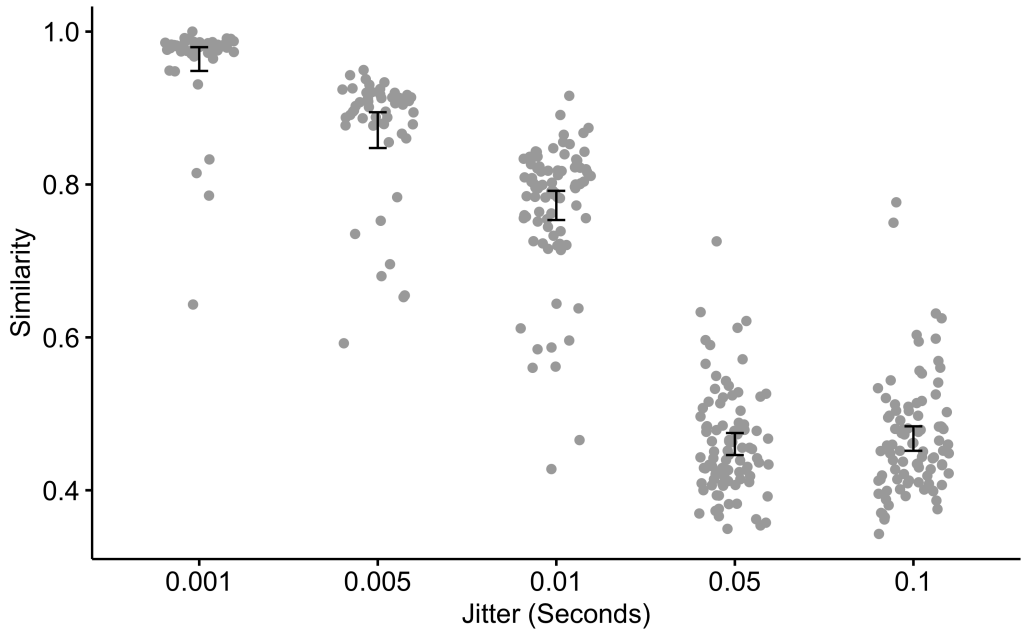


Figure 6.8: Similarity changing with artificially added start time jitter. The similarity level generally decreases as the jitter increases.

To investigate the effect of bin size and scale factor on similarity, I run the same self-querying process with pieces in the train set. Here, I dependently test with bin sizes [0.1, 0.5, 1, 5] (seconds) and scale factors [0.75, 0.9, 1.1, 1.25]. The musical effect of applying a scale factor is that the query is slowed down (scale factor < 1) or sped up (scale factor > 1). The scale factor is applied to the start time of all notes in a query, for every bin size above. Maximal similarity 1 can be obtained without applying scale factor (i.e., multiplying by scale factor 1). As Figure 6.9 shows, a difference in scale factor has little effect on similarity because *tdr* is robust to time scaling, whereas increasing bin size increases the similarity.[9] The latter result is an expected phenomenon, because, with a larger bin size, a false-positive hash entry that is located further away from the precise start time is more likely to be included by the approach.

Finally, I conduct a runtime test on building the lookup table, with ten datasets of increasing sizes. There are a total of 962 pieces in MAESTRO train set, the size of the subset ranges from 1 to 10 in multiples of 96.2 (ceiling function is applied). This test is run on Intel(R) Core(TM) i9-9920X CPU @ 3.50GHz and the runtime is recorded for plotting Figure 6.10, which contains two line plots for runtime against the number of notes and the number of hash entries created. The plot on the left shows that the runtime increases linearly as the number of notes increases. Although, according to the Algorithm 1, the time complexity for searching triples is $O(n^3)$, the actual runtime is linear to the size of input. I infer the difference is because the runtime for each piece

[9]Even if scale factor has a larger impact, it will be possible to iterate over several different likely scale factors during the matching process to take this into account, but the current analysis suggests this is not necessary, at least for the range of scale factors considered here.

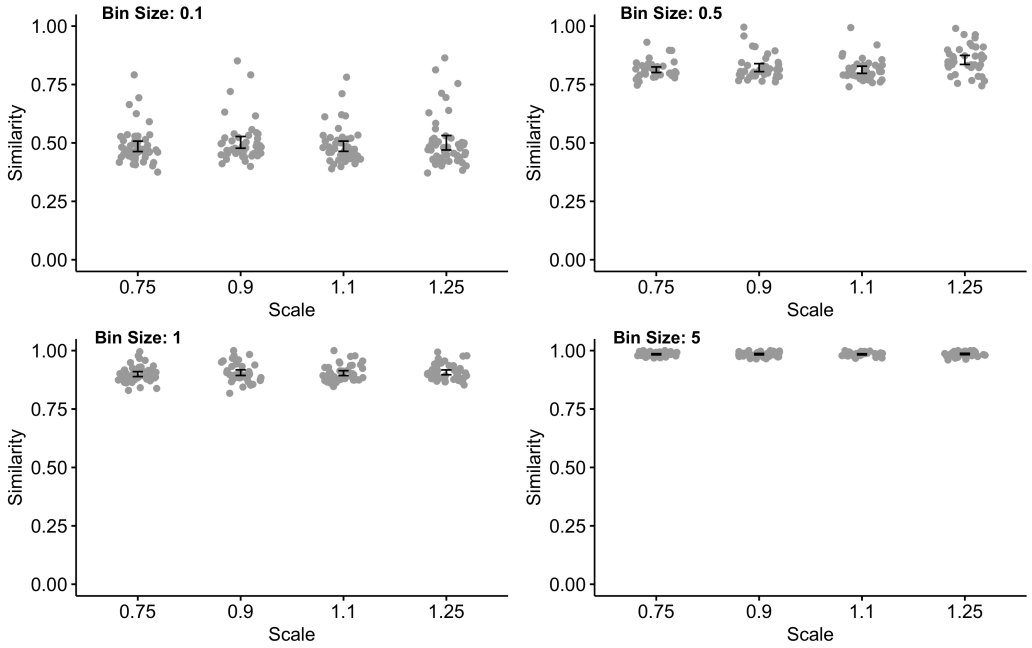


Figure 6.9: Similarity changing with a set of bin sizes [0.1, 0.5, 1, 5], and for each with a set of scale factors [0.75, 0.9, 1.1, 1.25].

is roughly the same, so the relationship becomes linear in this practical case. The plot on the right shows the runtime increases as the number of created entries increases, and combined with the plot on left, it indicates the space complexity is also linear.

6.6 Discussion

The work in this chapter addresses research question 1 and puts forward the notion that AI for music generation should result in outputs that imitate instead of merely copying original pieces, and highlights that checks of whether this is the case – what I refer to as the originality report – are often omitted. I introduce the methodology of the originality report for baselining and evaluating the extent to which a generative model copies from train-

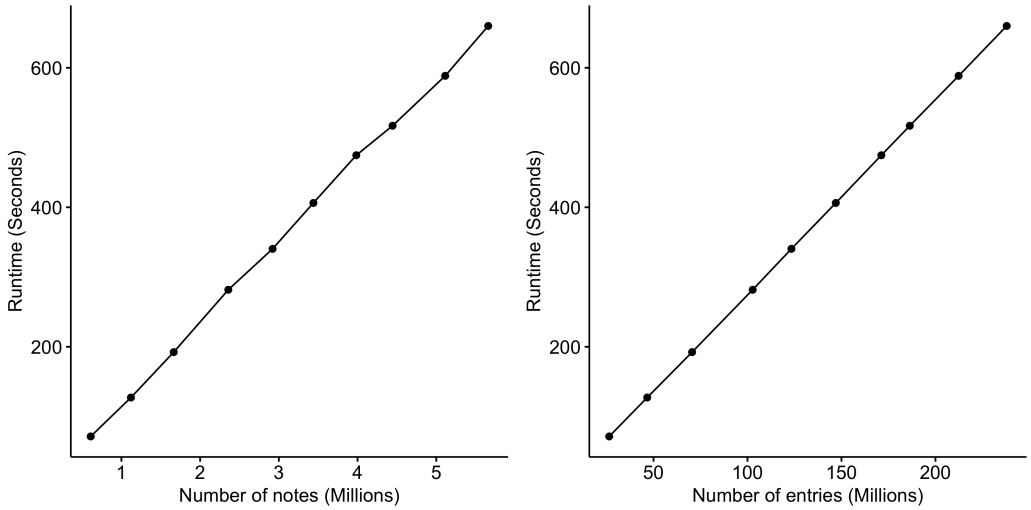


Figure 6.10: Two line plots of runtime in seconds against the number in millions of notes and entries created during lookup table building.

ing data. This originality report is parameterised by a particular similarity measure. Here I illustrate two similarity measures: a relatively straightforward note-counting approach based on the cardinality score (Ukkonen et al. 2003; Collins et al. 2014), and one based on fingerprinting score (Arzt et al. 2012; Collins et al. 2016a; Wang and Smith III 2012). By designing other similarity measures, it becomes possible to adapt the methodology to have emphases on different musical dimensions or aspects of those dimensions. I first use a relatively small dataset with rigid ontime information, and analyse outputs from two example models, one deep learning algorithm called Music Transformer (Huang et al. 2018) and one non-deep learning model called MAIA Markov (Collins and Laney 2017). For Music Transformer, I additionally analyse its output with the MAESTRO dataset (Hawthorne et al. 2019), which contains a larger amount of pieces and all with expressive timing information. The analyses illustrate the use of this methodology and the existence of music plagiarism in recent research.

I recognise Google Magenta for making their source code (e.g., for Music Transformer) publicly available, because it enables a level of scrutiny that has not always been possible for previous work in this field (Pachet et al. 2017). That said, the results indicate a phenomenon wherein this type of deep learning language model gradually copies increasingly distinctive chunks from pieces in the train set, calling into question whether it really *learns* to generate. Deep learning often requires large datasets with a wide variety to increase the model generalisation ability (Barbedo 2018). However, this does not necessarily increase the likelihood of generating outputs with a wide variety. Particularly here, these results highlight that the Music Transformer model's (Huang et al. 2018) is not sufficiently original compared to the self-originality of the corpus on which it is trained. Stepping back somewhat from these specific results, I see several findings in this paper (e.g., those associated with Figures 6.5(b), (d), (e), 6.6, and 6.7) as evidence that current deep learning models for music generation might be good "data memorisers and regurgitators", but fall short of being "original" or "creative" – not just qualitatively but as measured quantitatively by the originality report. More recent research finds the information in training data can be retrieved from large language models, which highlights various issues of memorization (Carlini et al. 2020). Furthermore, using the conventional stopping criteria for the training process, the "best" model not only has a low level of originality, but also the quality of generated excerpts is low in the sense that the same note is repeated most of the time (see Figure 6.5(d)). Going forward, the field of deep learning needs to reconsider in what situations the conventional stopping criteria are appropriate: perhaps loss and accuracy should

no longer be the only criteria when evaluating the model, because preventing these models from copying training data is needed, especially when they are used increasingly in a "black-box" manner by practising musicians.

6.6.1 Limitations

The size of the Classical string quartet dataset used above is smaller than that used for the original work on Music Transformer (Huang et al. 2018) – MAESTRO (Hawthorne et al. 2019) – and it is also quantised to a smaller set of time values. I address this issue by using MAESTRO also, but another potential solution is to pretrain a generative model with a large dataset to gain "general musical knowledge" and then finetune with a smaller style-specific dataset (Donahue et al. 2019; Conklin and Witten 1995). However, the first dataset still represents a substantial amount of Classical music, and certainly enough to give a human music student an idea of the intended style, so if deep learning algorithms cannot operate on datasets of this size, then it should be considered a weakness of the deep learning approach rather than a limitation of this methodology.

The simplicity of the cardinality score has some appeal, but as noted in the review of existing work, it can mean that subtle variations along some musical dimension destroy any translational equivalence, giving a low cardinality score that is at odds with high perceived similarity. For instance, the expressive timing in the MAESTRO dataset (Hawthorne et al. 2019) constitutes such subtle variations along the dimension of start time, and makes it ill-advised to use the cardinality score to assess the originality of an algorithm trained on these data. In addition, the cardinality score shares some

general advantages of the geometric approach. But in its current use, it is also not able to take into account the distinctiveness of excerpts being compared (Conklin 2010; Collins et al. 2016a). For instance, Figure 6.1 indicated an instance of similarity between Mozart and Haydn, but when taking into account how many Classical pieces end in this way, it is not a particularly distinctive or interesting example.

As mentioned, cardinality score does not work properly with subtle variations (e.g., expressive timing), while the symbolic fingerprinting using geometric hashing is able to detect similarities with such data. However, this complex approach also requires configuration of parameters (e.g., pitch/time difference range, bin size) to balance efficiency and effectiveness. For instance, increasing the pitch/time difference range extends the coverage of distinctive fingerprints, but this also exponentially increases space and time complexity. In terms of distinctiveness, I observe some fingerprints have higher frequencies than others in a dataset, which implies that highly frequent fingerprints correlate with the style of a music corpus, while infrequent ones make pieces distinctive from one another. However, the current version of similarity calculation, the ratio of unique fingerprints, does not reflect these distributional characteristics.

6.6.2 Future Work

I would like to see the originality report method that I have developed be embedded into the training processes of various music generation algorithms, to play a role as an advanced stopping criterion. Meanwhile, I will need to ensure that this criterion can still maintain the generalisability asserted by

standard stopping criteria.

I will investigate the compatibility of the originality report method with model selection, which is often conducted as an outer loop of model training. Loss function engineering is a topic addressed in recent novel generating strategies (e.g., Elgammal et al. 2017), so it should be possible to merge high/low originality scores as rewards/penalties in training loss, to further investigate the problem that I have identified of language-based deep learning models appearing to be little more than powerful memorisers.

I will also explore further alternative similarity measures. Weighting of shift errors (Collins et al. 2014) and consideration of distinctiveness (Conklin 2010) could both address limitations mentioned above arising from the simplicity of the cardinality score. I will optimise the time performance of the geometric hashing approach (Arzt et al. 2012), and investigate whether weighting the fingerprints based on their distribution in a dataset can make the fingerprinting score reflect distinctiveness. This should mean originality reports can be generated for an algorithm trained on *any* music data, taking into account distinctiveness with respect to an underlying corpus.

This chapter introduces an evaluation framework for measuring originality of a given collection of music against the other collection. In the next chapter, I aim to subjectively evaluate the performance of AMG systems on a wider range of musical dimensions by conducting a listening study.

7 Comparative evaluation

This chapter aims to address research question 2: "In the musicological aspect, what is the quality gap between 1) deep and non-deep learning music generation systems? 2) human composers and these AI-based generation systems?" As discussed in Chapter 6, a flaw of the originality report is that it cannot assess quality dimensions of music, which could lead to a case where an extreme random piece has high originality. In order to improve the evaluation method, this chapter evaluates the music generated by various music generation systems through a listening study, and verifies the hypotheses about the performance of these systems, via Bayes factor analyses of the rating data. The introduction of Bayes factor analyses is provided in Section 4.3.

7.1 Hypotheses

The listening study discussed in Section 7.4.3 consists of two parts differentiated by the target styles of the music therein: Classical string quartet (CSQ) and classical piano improvisation (CPI).[1] Each stimulus is rated according to six musical dimensions, defined in Section 7.4.1. Here I provide a list of

[1]The difference between Classical and classical is meaningful and is defined in Section 7.2.1.

hypotheses with respect to the performance of various systems for a subset of the styles and musical dimensions. These hypotheses stem from my understanding of the mechanisms behind the models used to generate stimuli for the listening study (see Section 7.2.2), and, as I prepared the stimuli for the study, their apparent strengths and weaknesses.[2]

7.1.1 System-focused hypotheses

1. *In the CSQ part of study, there will be no significant difference between the stylistic success ratings for MAIA Markov and Music Transformer.*

Both systems learn and generate music with the symbolic representation in a recurrent manner, that is statistically modelling the likelihood of the next state/event by the previous. Based on the generated results, I believe they share a similar strength in generating accurate notes locally. And I assume the local accuracy serves as an important factor of stylistic success when only symbolic representation is considered.

2. *In the CSQ part of the study, MAIA Markov will outperform other models on the repetitive structure rating.*

MAIA Markov forces the generated music to inherit structural patterns. It is supposed to make the repetitive patterns more significant than other systems. Although the multi-headed attention mechanism used in Music Transformer shows the efficiency and accuracy in capturing long-term dependencies, I deem the direct indication of repetitive structure to be more manifest than learning and generating a repetitive structure by chance.

[2]I pre-registered my hypotheses on the Open Science Framework so that I did not fall into the trap of devising them after seeing the listening study results. This trap, called post-hoc hypothesising, runs contrary to a scientific experimental method.

3. *In the CSQ part of the study, coupled recurrent model will get lower stylistic success ratings than MAIA Markov and Music Transformer.*

Through my listening experience, the generated excerpts from coupled recurrent model show less variety in pitch range and durations when compared to other models. Thus I doubt it to meet the expectations of Classical era.

7.1.2 Musical dimension-focused hypotheses

4. *Ratings of melodic success will be a driver/predictor of overall aesthetic pleasure.*

Melody, harmony and rhythm are commonly considered the most fundamental elements when analysing compositions. I aim to find out to what extent those elements are correlated with aesthetic pleasure. Intuitively, I think melody will correlate most strongly because it is the most memorable and representative element in these styles (Rosen 1997).

5. *Stylistic success ratings in most cases will be lower than aesthetic pleasure.*

Through listening to the generated excerpts, a short pleasant phrase can be often found locally, but it rarely persists within a coherent stylistic structure. Thus, I aim to verify this phenomenon with participants' ratings.

6. *Music Transformer will outperform other models on melody ratings in both CSQ and CPI parts.*

The weakness of generated music often stems from unexpected (negatively, instead of creatively) notes. I think this phenomenon largely affects melodic

aspects. With the prominent performance of multi-headed attention mechanisms, I believe that Music Transformer learns melodies more accurately than other systems.

7.1.3 Dataset- and participant-focused hypotheses

7. *Among systems taking part in both studies (including the Orig category), the stylistic success ratings received for CSQ will be higher than those for CPI.*

To my best understanding of CSQ and CPI, excerpts of CSQ are less stylistically diverse than those in CPI. Without models being specially adjusted to either style, excerpts generated in CSQ are more likely to be recognised by participants. Thus, they are supposed to receive higher ratings of stylistic success.

8. *Music Transformer will outperform MusicVAE on stylistic success ratings in the CPI part of study but not in the CSQ part.*

As mentioned above, I consider CSQ to be less stylistically diverse, and therefore statistically, the complexity of its dataset is lower than CPI's. With a similar model size and the assumption that Music Transformer is more powerful than MusicVAE, they should have the same performance in the CSQ part of study, but Music Transformer will show better adaptability with the CPI dataset.

9. *Participants with fewer years of musical training tend to give higher ratings.*

Particularly in stylistic success ratings, I assume that relatively expert judges are more sensitive to misplaced notes in an excerpt, while participants with fewer years of musical training are not as vigilant in capturing "wrong" notes. Thus, I suppose that participants with less formal musical training will evaluate music more based on intuition and less on rigorous analyses, and therefore will give higher ratings.

7.2 Systems under test

In this section, I describe two datasets and four AMG systems that I (re)-implemented to prepare the stimuli or excerpts indicated in Table 7.1. Supporting materials including stimuli, datasets and code for replicating the outputs can be accessed via `https://osf.io/96emr/`.

Table 7.1: Categories of stimuli contained in the CSQ and CPI parts of study. The first value indicates the number of stimuli generated per category; the second value, in parentheses, indicates the number of stimuli selected from the pool per category for each participant.

System name (abbreviation)	Classical string quartet (CSQ)	Classical piano improvisation (CPI)
Original (Orig)	25 (4)	25 (4)
Before-After (BeAf)	25 (4)	–
MAIA Markov (MaMa) (Collins and Laney 2017; Collins et al. 2016b)	25 (4)	–
Coupled Recurrent Model (CoRe) (Thickstun et al. 2019)	25 (4)	–
MusicVAE (MVAE) (Roberts et al. 2018)	25 (4)	25 (4)
Music Transformer (MuTr) (Huang et al. 2018)	25 (4)	25 (4)
Listen to Transformer (LiTr) (Huang et al. 2018)	–	25 (4)
Total	150 (24)	100 (16)

7.2.1 Datasets

So that the results here are not limited to a single musical style, I make use of two separate datasets: Classical string quartets and classical piano improvisation. As previously mentioned, while some AMG research operates in the audio domain, the focus here is on the symbolic domain. The Musical Instrument Digital Interface (MIDI) format is commonly used to describe musical symbolic data, from which features can be extracted and processed (see Section 2.2.1), such as attributes of musical notes: ontime, MIDI note number, duration, and velocity.[3] MIDI data may or may not comprise ex-

[3]Ontime is the start time of a note, counting in crotchet beats and taking 0 to be bar 1 beat 1 (Collins 2011, p. 347). MIDI note number is a numeric encoding of pitch, with

pressive timing and dynamics – it tends to depend on whether or not the data were captured by a human performer playing on a MIDI-enabled instrument. For example, the MAESTRO dataset (Hawthorne et al. 2019) consists of MIDI files collected from virtuosic piano performances, including multiple versions of the same piece with different expressive timing and dynamics. Humdrum (see KernScores[4] for example data) and MusicXML are other widely-used digital musical notations, which encode the original musical scores (sheet music). Timing and dynamic expressivity in these files can be absent entirely or limited to what the composer or editor marked on the score.

Classical string quartet

A Classical string quartet is a piece written for two violins, viola, and cello, following the style of Western art music written in the period between 1750–1830. This was a period during which composers Joseph Haydn, Wolfgang Amadeus Mozart, and Ludwig van Beethoven were active, and this music is often regarded as being of an "orderly nature, with qualities of clarity and balance, and emphasising formal beauty rather than emotional expression (which is not to say that emotion is lacking)" (Kennedy and Bourne 2004, pp. 171). My dataset contains 71 Classical string quartet movements, having a total of 228,021 notes, without expressive timing and dynamics in MIDI format from KernScores. The dataset was formed according to the following filters and constraints:

C4 or "middle C" being MIDI note number 60, C♯4 or D♭4 being 61, etc. Duration is the length of a note, again measured in crotchet beats. Velocity is the term used for the relative loudness of a note, with common ranges being $[0, 127]$ or $[0, 1]$.

[4] `http://kern.ccarh.org/`

- string quartet composed by Haydn, Mozart, or Beethoven;
- first movement;
- fast tempo, e.g., Moderato, Allegretto, Allegro, Vivace, or Presto.

Classical piano improvisation

A classical piano improvisation is a piece that usually would be created and played at the moment on the piano by someone who is able to draw on material from "classical" music to varying degrees of abstraction. I use the small "c" for "classical" which subsumes the "Classical" period 1750–1830, and in this study "classical" is taken to mean Western art music, following the conventions of music written in the period between 1650–1900, i.e., from the time of J. S. Bach to Brahms. I use the MAESTRO dataset (Hawthorne et al. 2019) for this part of study. It contains 1,276 virtuoso piano performances, with a total of 7.04 million notes, as MIDI data recorded with Yamaha Disklaviers. Unlike the MIDI data collected in KernScores, it contains expressive timing and dynamic.

I define this style based on the contents of the MAESTRO dataset because two of the deep learning models I want to investigate, MusicVAE (standing for variational auto-encoders) (Roberts et al. 2018) and Music Transformer (Huang et al. 2018), are introduced in papers that also use this dataset. To give these models the best chance of performing well in the listening study, I include the same dataset as used in the original study (Huang et al. 2018).

7.2.2 (Re)implementation of Models

MAIA Markov (Collins and Laney 2017) is the non-deep learning model included in this study. In its current form, this model can only be applied to non-expressive timing data, so it features in the CSQ but not the CPI part of the study. The model structure and state space are described in Section 3.2; I add here that the initial distribution is calculated by extracting estimated phrase beginnings and endings from the input data. Sometimes phrase beginnings and endings are encoded in kern files – transferred from phrase marks in the original score on which the encoding is based, but for the CSQs they are not, so I use rests of longer than one beat to identify phrase beginnings and endings. I use two repetitive structures that are common in Classical music: one where bars 1–4 are repeated in bars 5–8, and a second where additionally bars 1–2 occur again in bars 11–12. Thus, I assume MAIA Markov will perform well on ratings of repetitive structure.

A second model I include in the study, and the first example of a deep learning model, is the coupled recurrent model (Thickstun et al. 2019), which uses a hierarchical architecture based on recurrent neural networks to achieve polyphonic music generation. It consists of multiple sub-models for voices, and generates tokens in steps based on a certain historical window. There is a global state governing all voice generations, so a generated note in a single voice is not only conditioned by its voice history but also by the global state coordinating across voices. As such, the coupled recurrent model should perform well on ratings of harmony. I use the published scripts (multipart6 as reported as the best version, with the total loss of 12.87) provided with the original paper to generate CSQs. Again, this model is not applicable to

expressive data, so it is not included in the CPI part of the study.

The third model I include in the study is MusicVAE (Roberts et al. 2018). The assumption behind VAEs is that each data sample can be represented by a latent code with a smaller dimensional size, and all those latent codes lie in a predefined distribution (usually a normal distribution). During the training, the decoder is required to reconstruct the input data by using the latent code sampled from the distribution. On this basis, MusicVAE applies multi-layer bi-directional long short-term memory networks as encoder/decoder to compress/reconstruct sequential musical tokens. In addition, a conductor (as introduced in the original paper (Roberts et al. 2018)) is placed between the encoder and decoder to prevent posterior collapse, where the decoder breaks free of dependence on the encoder. Regarding the generation, the conductor firstly takes the randomly sampled latent code and produces sub-codes for lower level slices (e.g., bars/measures). Musical tokens are then recursively generated by decoders with each corresponding sub-code. In this listening study, I reimplement MusicVAE and generate excerpts for both CSQ and CPI part of study. The model consists of two encoder/conductor/decoder layers with a hidden size of 1,024, it was trained with scheduled sampling (Bengio et al. 2015) and the Adam optimiser (Kingma and Ba 2014). The loss for training MusicVAE is a weighted sum of cross-entropy and KL-divergence. The training starts with 20 warm-up epochs, in which the KL-divergence loss is multiplied by the weight that increases with the rate of 0.01 for each epoch. The trained model is then obtained by early stopping, with the best validation loss of 2.455 for CPI and 1.42 for CSQ.

Music Transformer (Huang et al. 2018) is said to benefit from the self-

attention mechanism, and therefore said to generate reasonably coherent polyphonic material on a short-term basis (several bars of music). The model is configured with six layers, eight heads, a model dimension of 512, and sequence length of 2,048. It was also trained with scheduled sampling (Bengio et al. 2015) and the Adam (Kingma and Ba 2014) optimiser, using cross-entropy loss with early stopping. As mentioned in Section 3.3, I reimplement Music Transformer achieving the validation loss of 2.2 for CPI and 1.184 for CSQ, and use it to generate excerpts for both the CSQ and CPI versions of the task. The original authors (Huang et al. 2018) also provide generated results, but these were trained on over 10,000 hours of piano recordings from YouTube, transcribed using Onsets and Frames (Hawthorne et al. 2018), which is not the same dataset as used in the original paper. So while a direct comparison between versions is not possible due to the different training sets involved, I randomly select some excerpts from "Listen to Transformer" and include them in the listening study as an additional category.[5]

7.3 Participants

For this study, I aim for the majority of participants to have a relatively high level of musical knowledge. While it is possible to achieve ostensibly impressive results with novice users or listeners (Louie et al. 2020), it does not benefit the broader purpose of measuring true progress (or lack thereof) in the field. One of the reasons I recruit relatively expert participants (e.g., music undergraduates) is because at this level of expertise, a listener can hear beyond the expressive inflections of a human (or algorithm) performance

[5] `https://magenta.github.io/listen-to-transformer`

and assess the stylistic success of the notes themselves. In other words, they can listen in a performance-invariant manner, and this is ecologically valid because when students write their own stylistic compositions for coursework, these are also assessed not on the basis of how or how well they are performed, but concerning the note data therein.

The listening study in this thesis has been approved by the Physical Sciences Ethics Committee of the University of York. A total of 50 participants were recruited, from email lists including music undergraduate students at the University of York, the Magenta Google Group, and the Society for Music Theory. Participants were compensated £10 for an hour of their time. After excluding all data from participants who went through the study so quickly it would have been impossible for them to listen to their assigned excerpts in entirety, 41 participants' submissions are used as the basis for the analysis in Section 7.5.

Participants' age (lower quartile = 20, median = 26, upper quartile = 37) against years of musical training (lower quartile = 9, median = 13, upper quartile = 18) is shown in Figure 7.1, and Table 7.2 shows participants' frequency of playing/singing (median = "Once a day") and attending concerts (median = "Once a month").

7.4 Experimental Design

Here I clarify the design decisions made to prevent possible bias in favour of or against particular systems. The selection of stimuli/excerpts in this listening study is performed based on the principles of balanced incomplete block design (BIBD) (Street and Street 1986). I aim to have each stimulus from

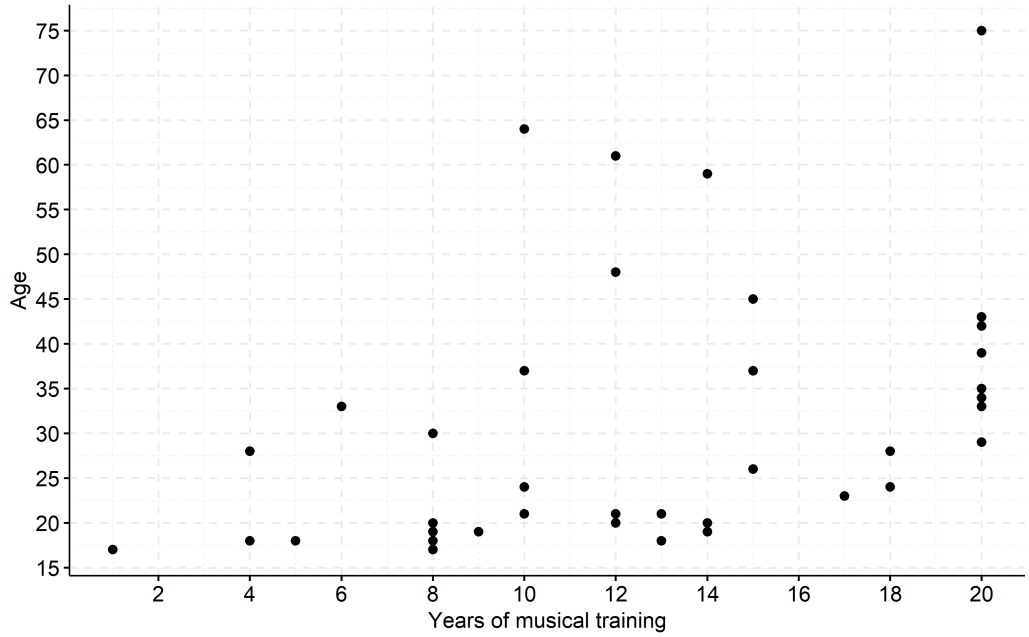


Figure 7.1: Participants' age against years of music training

Table 7.2: Participants' frequency of playing/singing and attending concerts

	Freq. of playing/singing	Freq. of attending concerts
Once a day	25	–
Once a week	8	4
Once every two weeks	3	7
Once a month	3	15
Once every three months	1	6
Once every six months	0	3
Once a year	0	2
Less than once a year	1	3
Never	–	1

the various systems be rated by sufficiently many participants, so that overall the results are unlikely to be biased towards arbitrary excerpts. Table 7.1 shows that there are 25 excerpts per system, giving a total of 250 excerpts in total across the CSQ and CPI versions of the listening task. Since it would be unreasonable, and lead to listener fatigue, to ask each participant to rate all of these excerpts, I sample from this excerpt pool to present a subset of excerpts for each participant, balanced with respect to blocks/categories. A naive sampling process may result in some excerpts being selected many more times than others. To mitigate this, I maintain a count of how many times each excerpt is presented as the study runs. Each participant is presented with 40 stimuli (four excerpts from each of the ten categories across both parts of study). I remove an excerpt from the pool when it has been presented to eight participants, and randomly select four from the remaining excerpts per category with each new participant. As reported above, I remove the data of participants who rush through the study, but still, the above steps help ensure a strong coverage of ratings for the excerpts from each system.

To prevent order effects, after an introduction page, participants are first redirected at random to either the CSQ or CPI version of the listening task, followed by the complementary version. Within task version, excerpts are shuffled to prevent ordering effects or associating particular groups of stimuli with particular systems.

Table 7.1 shows the four computational models introduced in Section 7.2.2, as well as "Listen to Transformer". From a design point of view, so that I can compare computational performance with human composers' capabilities, each task version also includes an *Orig* category consisting of excerpts

of human-composed music in the respective target style, the contents of which are detailed further in Section 7.4.2. For CSQ, I also make use of some excerpts of string music composed during eras on either side of the Classical period, so there is an additional *BeAf* category in this version of the task. This could enable me to shed some light on whether computer-generated excerpts are at least comparable in terms of stylistic success with out-of-period human-composed works (i.e., the computer models lack specificity of period but they do generate generally high-quality music), or if they fall short here also.

7.4.1 Definition of musical dimensions

Each excerpt is rated according to the following six musical dimensions, which are based on music theory (Rosen 1997) and previous work (Pearce and Wiggins 2007; Collins et al. 2016b):

Stylistic success (Ss): A stylistically successful excerpt can be defined as one that conforms, in a participant's opinion, to the characteristics of the definitions of CSQ and CPI and example excerpts, which were given on the introductory page of the study. (The example excerpts remained accessible throughout the study.)

Aesthetic pleasure (Ap): The extent to which someone finds beauty in something (DE CLERCQ 2019). The concepts of "stylistic success" and "aesthetic pleasure" are independent in the sense that two people may agree on certain characteristics of a music excerpt that make it stylistically successful, whereas they may disagree on the extent to which they personally like it.

Repetition or self-reference (Re): The reuse, in exact or inexact form, of musical material (e.g., notes, melody, harmony, rhythm) within a piece.

Melody (Me): A succession of notes, varying in pitch, which has an organised and recognisable shape. Melody is horizontal, i.e. the notes are heard consecutively.

Harmony (Ha): The simultaneous sounding of two or more notes; in this sense synonymous with chords. The organisation and arrangement of chords and their relationships to one another, vertically (at the same time) and horizontally (across time) over the course of a piece.

Rhythm (Rh): Everything pertaining to the time aspect of music (as distinct from the aspect of pitch), including event or note beginnings and endings, beats, accents, measures, and groupings of various kinds.

7.4.2 Stimuli

Here I describe the formation of stimuli (music excerpts) for the listening study. As mentioned at the beginning of Section 7.2, all study materials, including the stimuli and interface used, are available via an Open Science Framework repository, which can be consulted for confirming or supplementing the details provided below. I restrict all excerpts to a duration of 20–30 seconds, aiming to balance the length of the whole procedure while still providing a good number of excerpts to be tested. For the sake of comparison, participants hear all excerpts played back with high-quality piano sound samples. For CSQ excerpts, which did not have expressive timing or dynamics,

the velocity of each note is set to 0.8, in a range [0, 1] that modulates gain on an individual sample; for CPI excerpts, the expressive timing is retained and velocity was normalised using the minimum and maximum values found in the whole piece. The sounds for stimuli are generated at the moment using Tone.js (`https://tonejs.github.io/`). No effects are added, and no further normalisation is applied to the sampled audio files. The above decisions are in keeping with the playback functionality of desktop music notation software such as MuseScore. This is the key point here: the stimuli are ecologically valid in that they have a sound quality that participants are familiar with from their use of music software.

There are 150 stimuli in the pool for the CSQ part of study: 25 *Orig* excerpts from seven compositions by Haydn, Mozart, and Beethoven; 25 *BeAf* excerpts from five compositions by Vivaldi (before the Classical era) and Brahms (after the Classical era); 25 generated excerpts for each of the four systems under test; 25 from "Listen to Transformer". There are 100 stimuli in the CPI part of study: 25 original excerpts from five piano improvisations in the classical style[6]; 25 generated excerpts for each of the two systems under test; 25 from "Listen to Transformer".

In terms of human-composed excerpts (the categories *Orig* and *BeAf*), as the duration of a movement (contained in a MIDI file) is in most cases longer than three minutes, I am able to obtain three or four highly contrasting (independent if heard in isolation) excerpts by manually clipping the MIDI files. Thus, the extracted excerpts do not overlap one another; nor do they start or end in the middle of a note. This reduces the size of training set

[6]collected from `https://www.youtube.com/c/KyleLandryPianoTutorials/channels`

from 71 to 64 items, but I am also able to use these seven compositions from the CSQ *Orig* category as the validation set during model training. The seven are selected pseudo-randomly to give a balanced representation of the three composers (Haydn, Mozart, and Beethoven) and also of key signatures (mostly major keys).

When running the generation models described in Section 7.2.2, I notice some systems could generate poor-quality output containing one or more of (a) an incessantly repeated note or notes, (b) unusually long rests, (c) a chunk of noticeable copying from the training set Thus, I apply some automated filters to address (a), (b), and (c), as otherwise I would have been wasting participants' time with these excerpts. The filters exclude generated excerpts with a large number of repeated notes or long rests, and I then apply the originality report method (Yin et al. 2021, 2022) to measure and exclude excerpts that copy too much input in their output. As such, no hand- or cherry-picking of computer-generated output occurs when preparing the stimuli for my listening study. I do not generate outputs for *LiTr* (merely reused existing), but the researchers behind this model do explicitly state that the outputs from which I sample are "random, not cherry-picked samples".[7]

7.4.3 Procedure

During the listening study, participants go through a web-based questionnaire, listening to the prepared excerpts (see Section 7.4.2) and rating the corresponding musical dimensions using sliders (see Section 7.4). Partici-

[7] `https://magenta.github.io/listen-to-transformer/`

pants are not told which model generated the excerpt, but they do know that some excerpts are model-generated and some are composed by humans. They participate remotely, listening to the excerpts on their own machines with their own headphones or speakers. I asked participants to declare any vision or hearing problems that may impact their responses, but no such issues were reported.

The full procedure consists of an introductory page explaining the task and defining and giving examples of the CSQ and CPI musical styles, followed by (in random order) the CSQ and CPI versions of the tasks themselves, and finishing with a final thank-you page where the payment information is also issued. For each excerpt, participants are presented with the following instructions on a web form:

1. Rate the following musical dimensions on a scale of 1–7 (low–high):
    - stylistic success
    - aesthetic pleasure
    - repetition
    - melody
    - harmony
    - rhythm
2. Indicate time windows or instants, if a very low or high rating has been given to this excerpt and it was due to particular moments rather than an overall impression (optional)
3. A text box for any comments about the excerpt (optional)

7.5 Results

This section describes analyses of ratings obtained from the listening study and presents and discusses results of the Bayesian hypothesis tests. I use abbreviations for system names and musical dimension rating categories as stated in Table 7.1 and Section 7.4.1.

7.5.1 Overview

As a consequence of the number of usable datasets provided by participants and the categories and stimulus numbers (Table 7.1), there are a total of 1,640 sets of ratings, ranging from one to seven. To provide a general overview of the distribution of these ratings, Figure 7.2 shows violin plots for all six musical dimensions, for each system and style.

Before addressing the formal hypotheses from Section 4.3, I make some informal observations about the results in Figure 7.2. Ratings of *Orig* for both CSQ and CPI are higher for all dimensions when compared to generative systems, and *BeAf* has slightly lower ratings overall compared to *Orig*, but still higher than the generative systems. Comparing CSQ and CPI, excerpts generated in CSQ style appear to have higher ratings than those in CPI style.

Particularly for CPI, *LiTr* outperforms *MuTr* in all six musical aspects. Excerpts from both categories were generated by the Transformer Model, so such a large gap in performance raises some concerns as to why, which I will return to in Section 7.6.

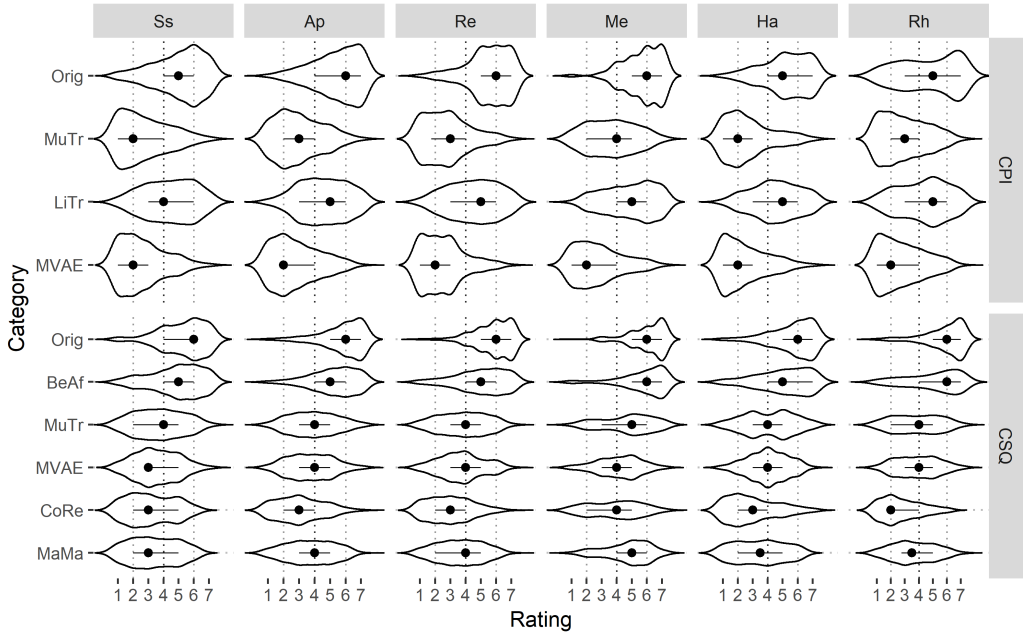


Figure 7.2: Rating (1–7) distributions of six musical dimensions: stylistic success (*Ss*), aesthetic pleasure (*Ap*), repetition (*Re*), melody (*Me*), harmony (*Ha*) and rhythm (*Rh*), for two styles Classical string quartets (CPI) and classical piano improvisations (CSQ) across different categories (mostly algorithms – see Table 7.1 for details). These show violin plots: the envelope shows the distribution of responses; the vertical lines are for reference to the rating scales; the horizontal line goes from the lower quartile, through the median (dot), to the upper quartile. (I do not use mean, variance, etc., as the ratings can only be considered ordinal values).

7.5.2 Hypothesis testing

To obtain a formal perspective on ratings and system performance, I conduct the hypothesis tests as introduced in Section 4.3, for each hypothesis listed in Section 7.1. Maintaining the same numbering of hypotheses, the following list indicates the outcome of each test, restates the null hypothesis H_0 and alternative hypothesis H_1, and also gives the computed Bayes factor BF_{10} and corresponding interpretation and degree of evidence according to Table 4.1. It should be noted that computing the Bayes factor with Gibbs sampling can lead to unstable estimates of extreme values, which means the variance of the Bayes factor becomes larger when the degree of evidence is more extreme (Lodewyckx et al. 2011). However, the large variance can be tolerated as it is unlikely to influence the overall interpretation.

System-focused hypotheses

1. *In the CSQ part of study, there will be no significant difference between the stylistic success ratings for MAIA Markov and Music Transformer.*

- Outcome: **There is no difference between *MaMa* and *MuTr* on *Ss* ratings.**
- $\mathbf{H_0}$: In CSQ, there is no difference between *MaMa* and *MuTr* on *Ss* ratings.
- $\mathbf{H_1}$: In CSQ, one of *MaMa* or *MuTr* outperforms the other on *Ss* ratings.
- $\mathbf{BF_{10}}$: 0.18 (moderate evidence for H_0).

2. *In the CSQ part of the study, MAIA Markov will outperform other models on the repetitive structure rating.*

- Outcome: **There is no difference between *MaMa* and *MuTr* on *Re* ratings; *MaMa* outperforms both *MVAE* and *CoRe* on *Re* ratings.**

- **H_0**: In CSQ, there is no difference between *MaMa* and (a) *MuTr*, (b) *MVAE*, (c) *CoRe* on *Re* ratings.

- **H_1**: In CSQ, *MaMa* outperforms (a) *MuTr*, (b) *MVAE*, (c) *CoRe* on *Re* ratings.

- **BF_{10}**: (a) 0.11 (moderate evidence for H_0); (b) 100.58 (very strong evidence for H_1); (c) 134.16 (extreme evidence for H_1).

3. *In the CSQ part of the study, coupled recurrent model will get lower stylistic success ratings than MAIA Markov and Music Transformer.*

- Outcome: ***MaMa* and *MuTr* outperform *CoRe* on *Ss* ratings.**

- **H_0**: In CSQ, *CoRe* shares the same performance with (a) *MuTr* and (b) *MaMa* on *Ss* ratings.

- **H_1**: In CSQ, (a) *MuTr* and (b) *MaMa* outperforms *CoRe* on *Ss* ratings.

- **BF_{10}**: (a) 2033.63; (b) 5492.46 (both extreme evidence for H_1).

Musical dimension-focused hypotheses

4. *Ratings of melodic success will be a driver/predictor of overall aesthetic pleasure.*

- Outcome: ***Me* ratings are positively correlated with *Ap* ratings.**
- **H_0**: *Me* ratings are not positively correlated with *Ap* ratings.
- **H_1**: *Me* ratings are positively correlated with *Ap* ratings.
- **BF_{10}**: $1.43e^{284}$ (extreme evidence for H_1).

5. *Stylistic success ratings in most cases will be lower than aesthetic pleasure.*

- Outcome: **Systems perform equally well on *Ss* and *Ap* ratings.** (Ratings were merged across categories for this test.)
- **H_0**: Systems perform equally well on *Ap* and *Ss* ratings.
- **H_1**: Systems performs better on *Ap* than *Ss* ratings.
- **BF_{10}**: 0.016 (very strong evidence for H_0).

6. *Music Transformer will outperform other models on melody ratings in both CSQ and CPI parts.*

- Outcome: ***MuTr* receives similar *Me* ratings as *MVAE* and *MaMa* for both CSQ and CPI parts of the study.**
- **H_0**: *MuTr* receives similar *Me* ratings as (a) *MVAE* in CSQ, (b) *MaMa* in CSQ and (c) *MVAE* in CPI.
- **H_1**: On *Me* ratings, *MuTr* outperforms all other systems in both parts of study.
- **BF_{10}**: (a) 0.213 (moderate evidence for H_0); (b) 0.433 (anecdotal evidence for H_0); (c) 0.554 (anecdotal evidence for H_0).

Dataset- and participant-focused hypotheses

7. *Among systems taking part in both studies (including the Orig category), the stylistic success ratings received for CSQ will be higher than those for CPI.*

- Outcome: **For systems involved in both CSQ and CSI parts of the study, *Ss* ratings are generally higher for CSQ.**
- $\mathbf{H_0}$: (a) *Orig* (b) *MuTr* (c) *MVAE* rate equally well on *Ss* for CSQ and CPI.
- $\mathbf{H_1}$: (a) *Orig* (b) *MuTr* (c) *MVAE* rate higher on *Ss* for CSQ.
- $\mathbf{BF_{10}}$: (a) 254.76 (b) 2247.95 (c) 37036.79 (all extreme evidence for H_1).

8. *Music Transformer will outperform MusicVAE on stylistic success ratings in the CPI part of study but not in the CSQ part.*

- Outcome: ***MuTr* and *MVAE* receive similar *Ss* ratings for both CSQ and CPI parts of the study.**
- $\mathbf{H_0}$: *MuTr* and *MVAE* have similar *Ss* ratings for (a) CPI and (b) CSQ.
- $\mathbf{H_1}$: *MuTr* outperforms *MVAE* on *Ss* ratings for (a) CPI and (b) CSQ.
- $\mathbf{BF_{10}}$: (a) 0.403 (anecdotal evidence for H_0); (b) 0.089 (strong evidence for H_0).

9. *Participants with fewer years of musical training tend to give higher ratings.*

- Outcome: **Years of musical training is not correlated with ratings given.**
- $\mathbf{H_0}$: There is no correlation between years of musical training and ratings.
- $\mathbf{H_1}$: Years of musical training are correlated with ratings.
- $\mathbf{BF_{10}}$: 0.059 (strong evidence for H_0).

Five of the nine outcomes above are as predicted in Section 7.1 (exceptions are #[5, 6, 8, 9]). The outcome of #1 suggests that *MaMa* and *MuTr* are capable of modelling the style of CSQ with the same performance. That is, the strongest-performing non-deep learning method *MaMa* is on a par with the strongest-performing deep learning method *MuTr*. Furthermore, *CoRe* shows inferior performance to *MaMa* according to #3. I can reasonably deduce that *deep learning methods have not surpassed non-deep learning methods in terms of their ability to automatically generate stylistically successful music.*

According to the outcome of #2, *MaMa* and *MuTr* have the same ability to generate repetitive structure. This is interesting because the two methods vary in the mechanism of pattern inheritance. *MaMa* uses a specified repetitive structure or one obtained via a pattern discovery algorithm (Collins 2011; Collins and Laney 2017) to guarantee that generated material inherits certain patterns, whereas *MuTr* achieves similar performance by relying on its representation and network architecture to capture relationships among notes. That is, over 30-second spans of music, what most music analysts would think of as short- or medium-term structure, deep learning methods

can achieve the same performance in generating repetitive structure through learning as a non-deep learning approach (Collins 2011; Collins and Laney 2017) achieves through explicit imposition of a structure. Whether such a finding applies at the long-term level, beyond 30-second spans of music, remains to be seen.

For hypothesis #3, *CoRe* is outperformed by both *MuTr* and *MaMa* on *Ss*. For hypothesis #7, systems are overall better at modelling the style of CSQ compared to CPI. For hypothesis #4, I verify that *Me* ratings are positively correlated with *Ap* ratings. Further tests with *Ha* ratings and *Rh* ratings reveal smaller positive correlations.

Hypotheses #[5, 6, 8, 9] do not result as expected. In frequentist statistics, all I would be able to say is that I have failed to refute the null hypothesis. Because I am using a Bayesean approach, I can make stronger claims.

For hypothesis #5 the intention is to investigate whether systems generate generally listenable music but fail to capture certain stylistic aspects. The results indicate that generally this is not the case, and systems have similar performance on *Ss* and *Ap* ratings in both CSQ and CPI.

For hypothesis #8, I compare *MuTr* and *MVAE* on *Ss* ratings separately in CSQ and CPI. I predicted similar performance in CSQ but superior performance for *MuTr* in CPI, since CPI is a more diverse, complex style to model and *MuTr* has a more powerful architecture than *MVAE*. The results however indicate that the systems have similar performance to each other in both styles. The evidence for similar performance is only anecdotal for CPI, whereas it is strong for CSQ.

The hypothesis and outcome of #6 is somewhat similar to #8, but here

specifically for *Me*: I predicted but do not see in the statistics evidence for *MuTr* having superior performance. In two cases (versus *MaMa* in CSQ, and versus *MVAE* in CPI) the evidence was again only anecdotal. The anecdotal results in #6 and #8 may warrant following up in future work.

The results for hypothesis #9 indicate there is no significant correlation between years of musical training and ratings given. This may be due to range restriction: I recruit only relatively expert listeners, so did not sample from a less expert pool of listeners who are sometimes more generous with stylistic success ratings in such listening studies.

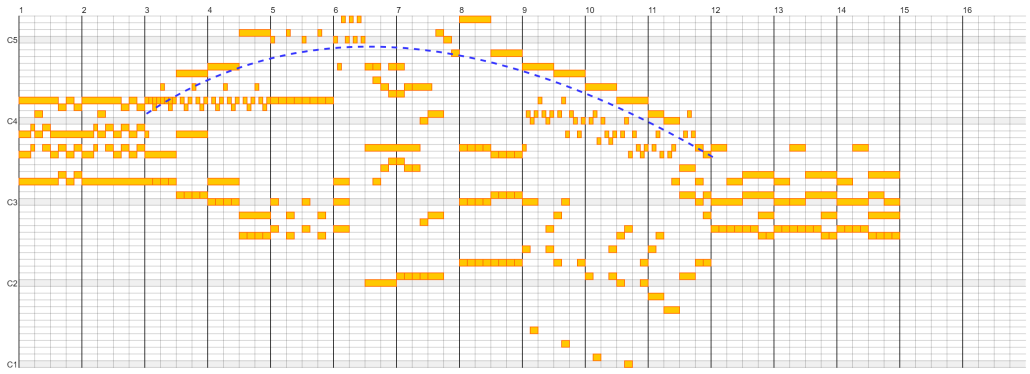
7.5.3 Musicological analysis

Here I provide and comment on piano-roll (pitch against time) representations of some stimuli that warrant particular attention due to their reception in the listening study[8]. As a complement to the statistical analysis, I analyse some of the open-ended comments received too. Commenting and highlighting sections of stimuli is optional in the listening study; 305 out of a total 1,640 submitted responses have comments. I choose piano roll over staff notation for the following figures as the former is more interpretable for general readers.

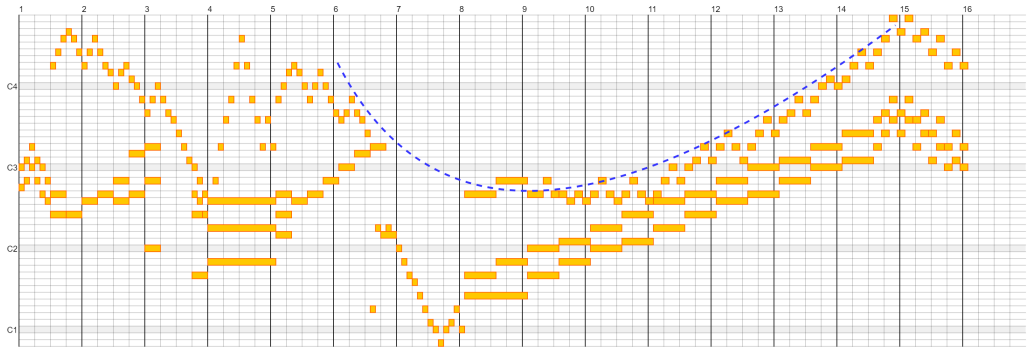
Figure 7.3 shows three excerpts from the CSQ part of the study that received the highest median stylistic success ratings in their respective categories: (a) a sample of *Orig* composed by Haydn[9]; (b) an excerpt generated by *MuTr*; (c) an excerpt generated by *MaMa*. The human-composed excerpt in Figure 7.3(a) has a clear melodic arc indicated by the blue dashed

[8]Excerpts can be accessed via `https://osf.io/96emr/`

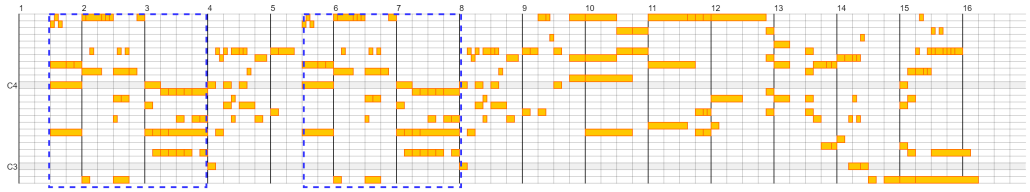
[9]Haydn, Quartet in F Minor, op.20 no.5, first movement, bars 35–48.



(a) *Orig*



(b) *MuTr*



(c) *MaMa*

Figure 7.3: Excerpts in CSQ with the highest median *Ss* ratings in their respective categories.

curve: the upper line rises to its highest point in bars 6–8 and falls to a lower register in bars 12–14. A sequential progression can be identified in bars 9–10.[10] The arc and sequence mechanisms are important for creating a sense of narrative or purpose and self-referentiality in music, but it is something rarely demonstrated in model-generated music. Self-reference or repetitive structure can be recognised in multiple places in Figure 7.3(a): bars 1 and 2 are very similar to one another; so are bars 12, 13, and 14.

The arc indicated by blue dashed curve in Figure 7.3(b) is inverted compared to that of (a): It begins and ends at relatively high pitches with a dip in the middle, which is less common in music of this period than the other way around, but is potentially preferable to no arc at all. The latter half (bars 8–15) contains ascending, sequential material. While there is no medium-term repetitive structure, bars 1.5–2.5 are similar to bar 5. That said, there is an unusual rhythmic displacement where the long chord in bar 4 ends one quarter of the way through the first beat in bar 5. I deduce that the serialisation method used by MuTr makes the generated music vulnerable in aspects of rhythmic coherence. Some participants think (b) starts off well, but the "wonky" harmony means the latter part is perceived as stylistically incoherent.

The model-generated excerpt in Figure 7.3(c) also demonstrates a repetitive structure, with bars 1–4 repeating in bars 5–8, marked by blue dashed boxes, but it lacks the arc of either (a) or (b).

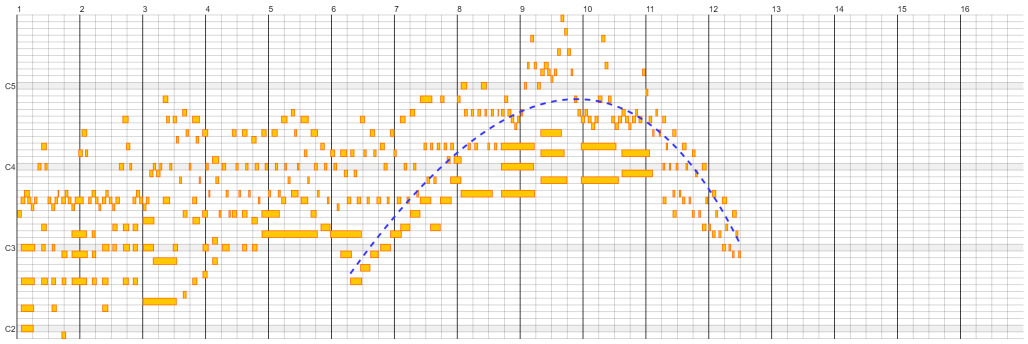
Figure 7.4 shows three excerpts from CPI part of the study that received the highest median stylistic success ratings in their respective categories:

[10] A sequence in music is when a collection of (pitch, rhythm)-pairs is reused by shifting up or down in pitch and later in time by a fixed amount.

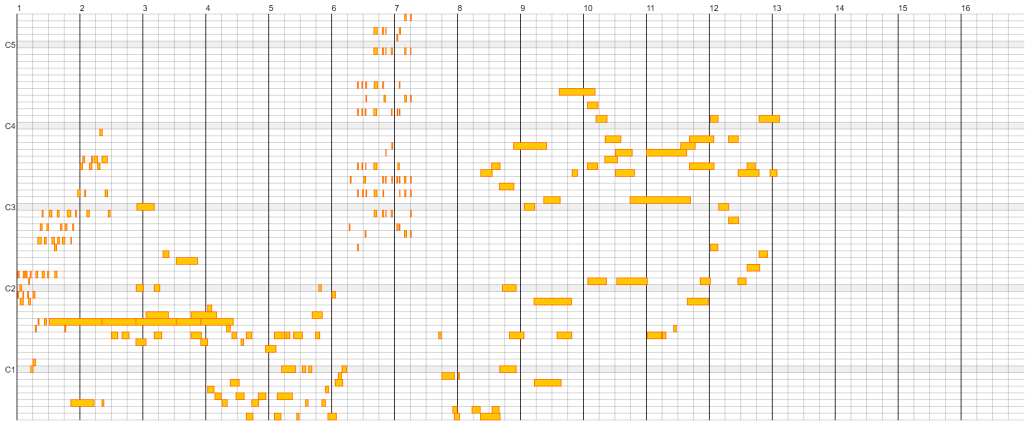
(a) a sample from *Orig*, (b) an excerpt generated by *MVAE*, and (c) an excerpt generated by *MuTr*. Compared with CSQ, excerpts in CPI contain more short notes resulting in rapid rhythms. Figure 7.4(a) contains a motif during the first half of bar 1 that is repeated three times with variations of ascending pitches. The same motif is again apparent in bar 10. The excerpt as a whole contains an arc, particularly from bar 6 to the end. Figure 7.4(b) shows evidence of some local arcs, but the overall shape is less recognisable. Significant repetitive structures cannot be found, and the sudden rapid chords in bars 6–7 undermine the preceding coherence, although naturally a balance between expected and unexpected musical events needs to be struck. One participant observes that the sixteenth notes in the beginning half "betrayed" the rest of the piece, as they make for an odd pairing. The melody from bar 8 to the end is pleasant yet seems unfinished. Figure 7.4(c) demonstrates repetitions in various places like bars 4–6 and 8–10, which focus more on single note repetitions instead of phrases as with other excerpts. It does not have a clear arc either: some participants think it is inconsistent with respect to practices of the classical period but compelling to listen to nonetheless.

The excerpt in Figure 7.5 attracted some contradictory comments. Some participants indicated this excerpt contains too many repetitions, while others appreciated the crossing of the voices, and the overall sonic effect. Objectively, it highlights the ability of *MuTr* to generate excerpts with clear, identifiable voices, just as in the target style of Classical string quartets.

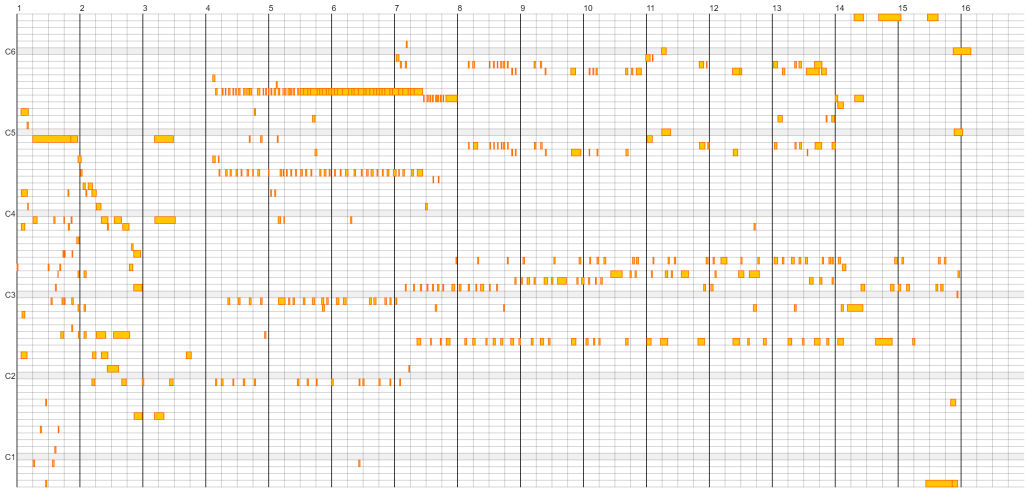
Deep learning systems suffer from copying large chunks of the training set in their outputs (Yin et al. 2021, 2022). This is the case here. The stimuli of *LiTr* are randomly selected from the web radio station published



(a) *Orig*



(b) *MVAE*



(c) *MuTr*

Figure 7.4: Excerpts in CPI with the highest median *Ss* ratings in their respective categories.

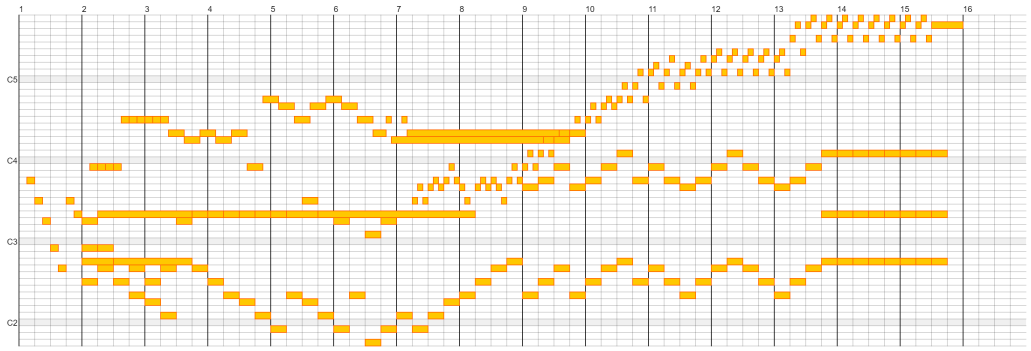


Figure 7.5: An example of *MuTr* in CSQ

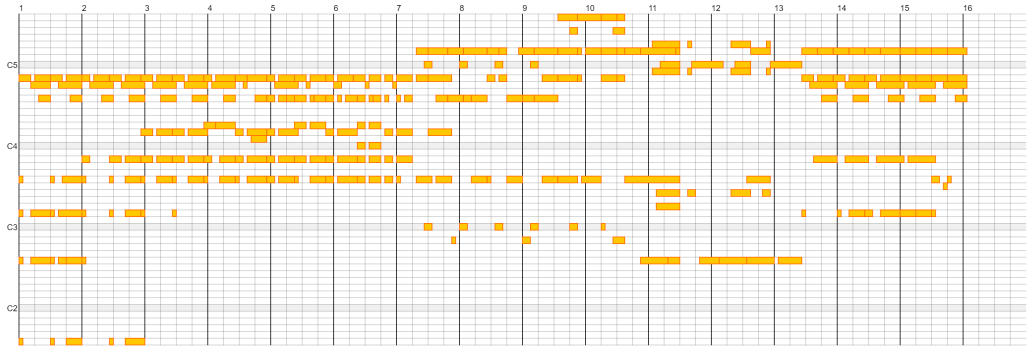


Figure 7.6: An excerpt of *LiTr* found mimicking "Carol of the Bells"

by the Magenta research group. Some of participants recognised an excerpt from *LiTr* as shown in Figure 7.6 copying the motif of "Carol of the Bells" composed by Mykola Leontovych (1914). This underlines the need for a systematic and widely adopted method to identify copying and potential copyright infringement in the output of AMG systems, for example, the originality report of (Yin et al. 2021, 2022).

7.6 Discussion

Deep learning models have come to prominence and have been recognised, due to rigorous comparative evaluation, as state-of-the-art solutions for many

tasks in machine learning, particularly in computer vision and natural language processing. Prior to this work, no such comparative evaluation existed for AMG, where the research effort devoted to evaluation and evaluation methodologies has fallen behind the efforts invested in training and generating with novel network architectures (Pearce and Wiggins 2001; Jordanous 2012; Yang and Lerch 2020).

7.6.1 Findings

To address research question 2, this work reports the results of a comparative evaluation involving appropriately expert judges, for various symbolic AMG systems as well as human-composed material. This comparative evaluation covers four AMG systems: one non-deep learning method (Collins and Laney 2017), and three deep learning methods with different generation strategies (Huang et al. 2018; Roberts et al. 2018; Thickstun et al. 2019). I use these systems to learn from datasets and generate them in two target styles: Classical string quartets and classical piano improvisations. Combined with human-composed excerpts, I have conducted a listening study that asks participants to rate excerpts according to six musical dimensions on a scale 1–7. I analyse the results in the context of a non-parametric Bayesian hypothesis testing framework.

No advances due to deep learning, and copying problems

Broadly, the listening study results demonstrate that the assumption of superiority of deep learning methods for polyphonic AMG is unwarranted. The non-deep-learning model MAIA Markov (Collins and Laney 2017) had the

same level of performance as that of the best-performing deep learning models, MusicVAE (Roberts et al. 2018) and Music Transformer (Huang et al. 2018). Some might say the comparison is unfair because MAIA Markov incorporates repetitive structure explicitly, whereas MusicVAE and Music Transformer do not. But the titles of the MusicVAE and Music Transformer papers are "A hierarchical latent vector model for learning **long-term structure in music**" (Roberts et al. 2018) and "Music transformer: Generating **music with long-term structure**" (Huang et al. 2018) (my emphases), suggesting that the deep neural networks they employ are capable of a) modelling long-term structure in music, and consequently b) generating outputs where this long-term structure is evident/perceivable. As such, my comparison and conclusion are fair. I also find that a large gap still exists between the stylistic success ratings for the strongest-performing computational models and the human-composed excerpts. Deep learning methods may be improving relative to one another according to metric-only evaluations, but these metrics appear to be of limited use, in that the listening experience of my participants indicates there is no improvement in stylistic success beyond a non-deep-learning method, and there is still an obvious gap in ratings compared to human-composed music.

In recent years, artificial intelligence systems have been increasingly involved in music creation and production, with musicians using them as cues for ideation, harmonisation, synthesis, mixing, and so on. An ethical issue here is that the data-driven property of generative models can lead to those systems copying chunks from original music data, which may result in plagiarism or copyright infringement. According to my previous work (Chapter 6,

Yin et al. 2021, 2022), existing deep learning AMG research neglects to check copying as part of its evaluation process. In that work, I discuss the nature of the language model-based Music Transformer algorithm, and show the extent of replication through various training checkpoints. For the listening study, I apply the same method (Chapter 6, Yin et al. 2021, 2022) to measure and exclude excerpts that copy too much from the training set. The originality baseline, showing the percentage of copying, is constructed with training and validation sets, and any generated excerpt with originality less than the lower 95%-confidence interval of the baseline is removed. In choosing 25 generated excerpts for each category of CSQ, 44% of MusicVAE (Roberts et al. 2018) excerpts and 56% of Music Transformer (Huang et al. 2018) excerpts had to be removed due to the copying issue, and new excerpts generated until 25 passed the originality test. On the other hand, no over-copying was identified in excerpts generated by MAIA Markov (Collins and Laney 2017) or the coupled recurrent model (Thickstun et al. 2019). With the "Carol of the Bells" example mentioned above in relation to Figure 7.6, Music Transformer is again shown to exceed the level of borrowing considered reasonable among human composers. Although quantifying music originality is still a challenge for the development of AI systems and musicology, efforts are being made in the areas of plagiarism detection and benchmarking. For example, Spotify recently applied for a patent for its AI-driven music plagiarism detection method, based on a method for sequence alignment from the 1980s (Pachet and Roy 2020; Smith and Waterman 1981).

Taking these two outcomes together – a non-deep-learning method (MAIA Markov (Collins and Laney 2017)) performing comparably to the strongest

deep learning method (Music Transformer (Huang et al. 2018)) in a human listening study where the participants are uninformed of the source of the composition, and the deep learning method raising concerns of ethical violations from direct copying – it seems there is still much for deep learning researchers to consider, improve upon, and evaluate, before a deep learning approach can be considered superior for AMG.

Gulf between 'Listen to Transformer' and 'Music Transformer'

There is a gulf between *LiTr* (Listen to Transformer) and *MuTr* (Music Transformer) across all rating categories. Most likely, this is due to differences in training data: *MuTr* is trained with the MAESTRO dataset (Hawthorne et al. 2019) while *LiTr* is trained with a wider range of piano datasets: this is evidenced by the generated excerpt that copies "Carol of the Bells" shown in Figure 7.6, which is not included in the MAESTRO dataset and is not in the "classical" era.

I do not know the stopping criteria for training used for *LiTr*, and differences in these between *MuTr* and *LiTr* could also lead to differences in the results. The reason I used early stopping for my reimplementation is because it replicates the criteria stated in the original Music Transformer paper. Moreover, my previous work (Yin et al. 2021, 2022) demonstrated that systems can even start overfitting (in calculation of originality with respect to the training set) before the epoch with the lowest validation loss. This prompts the question: is early stopping still enough for generative models, where the results usually need to be further assessed by other criteria (e.g., originality, musical metrics)?

7.6.2 Limitations

While I am confident in the functional equivalence between my reimplementation of *MuTr* and the original (Huang et al. 2018), I accept it is a slight but unavoidable limitation of this work that it has not been possible for me (or other researchers) to use the original implementation due to dependency issues.

The listening study highlights that the performance of AMG systems can vary according to target style and/or corresponding dataset (outcome of hypothesis #7). To prevent the case where a certain system is optimised for one specific style of dataset, I test with two target styles: Classical string quartets and classical piano improvisations. However, the coverage of musical styles is still narrow and does not provide a broad basis for claiming my findings apply in general. I observe that the system performance is usually better for CSQ than for CPI. Based on the fixed configuration of model training, I believe that the difference is caused by the stylistic complexity of the compositions in each style: CSQ is narrower in style than CPI, and the music in CSQ shows more regularity which makes composition rules more predictable.

As described in Section 7.4.2, I apply some filters for repeated notes, long rests, and the originality report (Yin et al. 2021, 2022) to remove excerpts that copy large chunks of the original training data. Investigating the extent to which the system can be creative, in the sense that it is producing a novel output from its input, is another important aspect of creativity system evaluation (Jordanous 2019) that is beyond the scope of this study. The 2021 version of the originality report supports only music data with non-expressive

timing. Therefore, I applied it only to outputs from the CSQ dataset.

The selected musical dimensions for evaluation are obtained from prominent writings on Western music (Rosen 1997). The evaluation framework here may not be applicable to other music styles (for example, atonal music or non-Western music), as it is not in their nature to emphasise these same musical dimensions.

In choosing to include both a model that performs poorly (Coupled Recurrent Model, (Thickstun et al. 2019)) and human-composed excerpts, I may have made it more difficult for listeners and, consequently, my statistical analyses, to distinguish differences between middling-performance systems. However, it is important to include a number of computational models in the study as well as human-composed music.

7.6.3 Future Work

Feature extraction is a common approach in the field of music information retrieval, where dimensions of music are summarised or quantified (McKay et al. 2018). This could be used to further explore and analyse the relationship between musical components and the various ratings I gathered. The aim of such work would be to identify features that have explanatory power in terms of predicting relatively low or high ratings for generated music, and to help assess the capability of generative systems to model creativity.

The term "style" refers to (at least) two concepts in music: musical style, meaning the period in which a piece of music was written (influencing the choices of which notes to write and how to combine them – crudely, the note data); performance style, meaning the way in which it is performed (char-

acterising the expressive performance parameters – crudely, adjustments in start times and durations of notes, and how strongly/softly they are played). As such, the former is mostly the domain of the composer, while the latter is mostly the domain of the performer. This is an oversimplification because artists will sometimes take a classical piece, arrange it (change or add notes, instrumentation), and play it in, say, a jazz style. But still, the point remains that musical style and performance style are identifiable, separable concepts. Accordingly, and prior to the arrival of deep learning models for music, researchers tended to investigate these two topics separately. For instance, Conklin and Witten (1995) and Collins et al. (2016b) investigate musical style, while Widmer (2002) and Grachten and Widmer (2011) investigate performance style. When Google Magenta started publishing on AMG models including Roberts et al. (2018); Huang et al. (2018), they took the rather bold step of attempting to model both musical style and performance style all in one neural network architecture. Given the conclusions drawn from the current study, it might be advisable to separate out these concepts again in future work, at least until the note-level originality issues of deep learning AMG models can be better understood and addressed.

As mentioned before, the field of AMG lacks a comprehensive and standardised evaluation framework, although I claim the current paper goes some way towards filling that gap. In future work, I will use the collected rating data and extracted features to predict ratings for new, previously unheard musical excerpts. As such, the predictive model would mimic the evaluation criteria of human listeners, have some validity for the target styles of CSQ or CPI, and so remove the need for conducting a new listening study every time

a researcher wants to evaluate a new network architecture or minor modifications to an existing model. This predictive approach would contribute to a greater degree of standardisation of evaluation procedures for AMG, as well as make systems more directly comparable. I will also consider integrating the distributional comparison approach of (Yang and Lerch 2020) into my evaluation framework.

Considering the amount of data used for both parts of the study, in CSQ I used a relatively small dataset compared to CPI. Although the CSQ dataset here represents a substantial amount of Classical music, and certainly enough for a human with general music knowledge to have an idea of the intended style, I would like to follow up on this principle by codifying it, pre-training the model with a large, generic music dataset, which gives the model "general musical knowledge", and then fine-tuning it with a smaller dataset in the target style (Donahue et al. 2019; Conklin and Witten 1995). This would be a more realistic model of how humans learn to compose, and so I can imagine it may lead to a higher quality of generated outputs from computational models.

In Section 7.5.3, I observe that several generated excerpts do not exhibit a clear arc. While such a shape is not a necessary property for stylistic success, it is at least one that can be perceived from a piano-roll plot, and that could be quantified in future. I also suggest that the slicing method used during data preprocessing could be altered in this regard: slicing a whole piece into sub-sequences of fixed size may result in a large portion of excerpts not having a clear arc, so that the model cannot learn the abstract/high-level pattern because it is not present sufficiently often. In future, I could use certain

segmentation methods (Rafael et al. 2009; Rodríguez-López and Volk 2013) that might cause the training excerpts to have more appropriate beginnings, middles, and ends with respect to learning abstract patterns.

7.6.4 Conclusion

This chapter presents a comparative evaluation of four symbolic AMG systems. To the best of my knowledge, there was no such evaluation comparing deep learning-based and Markov-based approaches prior to this work. The evaluation here is based on a human listening study and hypothesis testing with a Bayesian method (van Doorn et al. 2020), which contributes to filling a gap in the comparative evaluation of AMG systems, and the wider uptake of Bayesian methods across the evaluation of machine learning systems. The results show that the best deep learning and Markov-based algorithms for AMG perform equally well, and there is still a significant gap to bridge between stylistic success ratings received by the strongest computational models and human-composed music.

This checklist of suggestions for methodological fortitude is intended to help recover and sustain the accurate measure of progress in the field of AMG, as well as more broadly in the application of machine learning to domain-specific tasks:

1. Conduct a comparative evaluation via a listening (more broadly, participant) study;

2. If your particular technique or approach is X (neural networks, say), include at least one system from beyond X that can also be used to

address the same task and that the literature suggests will be a competitive point of comparison;

3. Having formulated hypotheses about the likely findings of the study, pre-register (state) them via a service such as Open Science Framework;

4. Detail how you recruited and compensated your participants, as well as describing their musical (more generally, domain-specific) backgrounds and level of expertise;

5. Publish the listening study interface and associated stimuli so that other researchers can attempt to replicate your findings;

6. If you collect Likert ratings, use non-parametric statistics to analyse them; use Bayesian rather than frequentist hypothesis testing to evaluate your hypotheses;

7. If you insist on publishing via arXiv or blog (for time-stamping or marketing purposes, respectively), then follow up with submission of the work to a peer-reviewed conference or journal.

This chapter provides a subjective evaluation of the selected AMG systems. It comprises proposing hypotheses about the performance of systems with respect to six musical dimensions, collecting ratings of each system in each dimension through a listening study, and verifying the proposed hypotheses by using Bayes factor analyses. To bridge the subjective ratings and objective features, the next chapter presents several types of quantitative features that can be extracted from a single or set of music pieces, and then I investigate the correlation between these features and the collected ratings in this chapter.

8

Musical feature extraction

This chapter aims to address research question 3: "What musical features can be extracted from music data to represent the quality of various musical dimensions?" Musical features can be used to describe the characteristics of a corpus (Eerola and Toiviainen 2001), such as the distribution of pitch classes, durations and intervals, these distributions often vary by music style. Yang and Lerch (2020) extract various pitch and rhythm-based features, and calculate their Kullback–Leibler divergence between the generated samples and the training dataset to evaluate AMG systems. Some AMG works also evaluate the generation performance by investigating the statistics of musical features extracted from the outputs (Sturm and Ben-Tal 2017; Dong et al. 2018). Besides the feature extract in symbolic domain, acoustic features are also widely used as model input for music emotion recognition (Zhang et al. 2016) and style classification (Nanni et al. 2016).

Returning to symbolic domain, there are two commonly used tools for musical feature extraction: music21 (Cuthbert et al. 2011) and jSymbolic (McKay et al. 2018). However, as mentioned in Chapter 4, these extracted features are most fundamental and low-level (Statistics calculated using only a single factor), and cannot reflect the high-level properties of music (e.g., structure and complexity). To address research question 3, I present several higher-

level features and investigate their correlation to the ratings collected in Chapter 7.

8.1 Features

Here I introduce six features that can be extracted from a single piece or a corpus, each feature aims to measure a certain aspect of music, for example, music structure complexity, melody arc shape and tonality. With these feature values, I investigate their relation to the collected ratings in each music dimension.

8.1.1 Statistical complexity

The quantitative or analytical definition of complexity varies by context. In measuring the randomness, Kolmogorov (1965) defines algorithmic information content as the length of the shortest program which causes a standard universal Turing machine to produce the exact string as output. Bennett (1988) defines logical depth as the run time required by a standard universal Turing machine to generate an object from an algorithmically random input. Both measures are independent from the actual dynamical system or process. However, describing an object from complete order to disorder is often insufficient, the sequence of musical notes in a piece of music is ordered, but the accumulation of order does not degenerate into mere repetition, the novelty of the composition is always reflected. Crutchfield (1994) categorises the above measures of randomness as deterministic complexity, and introduces a new type called statistical complexity $C_\mu(x)$, which describes the minimum

amount of historical information required to make optimal predictions of x at Shannon entropy rate h_μ. The statistical complexity of both ideal random object and simple periodic processes is defined to be zero as shown in Figure 8.1.

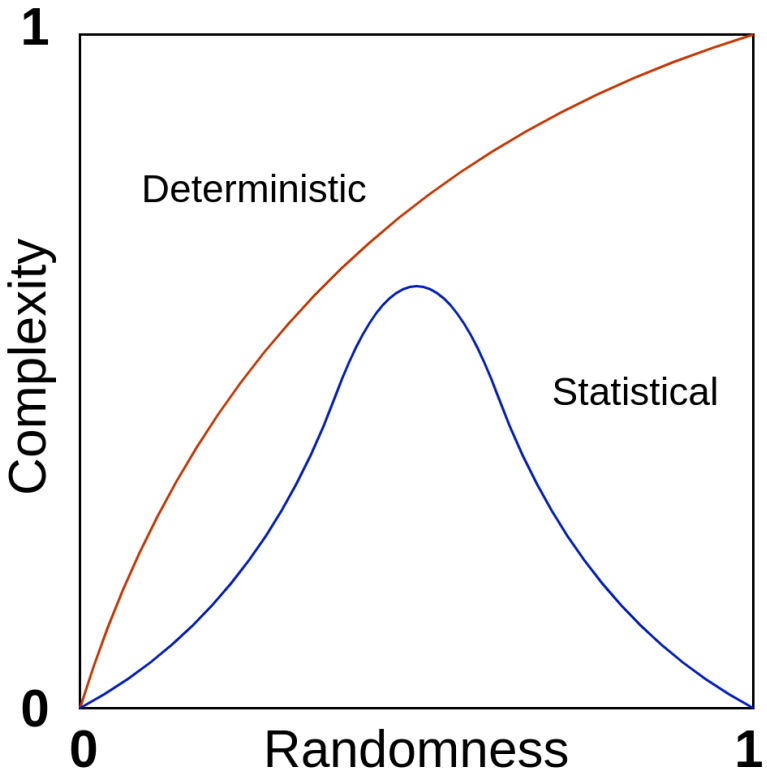


Figure 8.1: The red line indicates deterministic complexity, which measures the degree of randomness with Shannon entropy h_μ; the blue line indicates statistical complexity, where the ideal randomness is considered statistically simple, so the complexity of a predictable yet stochastic process increases.

Statistical complexity can be directly calculated from ensembles of strings, where a string is formed by consecutive symbols. The calculation starts with clustering sequences of symbols into states, where each state has its probability of emitting the symbols, and if sequences have identical conditional probability of any future symbol, these sequences are considered to be in

the same causal state. Once all causal states have been identified, the transition probabilities between causal states can be extracted from the data, and a probability distribution over all causal states can then be obtained. The statistical complexity is defined as the Shannon entropy of the states. Causal-State Splitting Reconstruction (CSSR) (Shalizi and Shalizi 2004) is an algorithm used to estimate causal states and calculate statistical complexity from data, I implemented CSSR with its public source code.[1]

As shown in Section 2.3, music can be represented by various sequential forms. Here I convert music data into sequences of tokens with the serialisation method described in Section 2.3.2, CSSR algorithm is then applied to these sequences to obtain the statistical complexity. An example sequence corresponding to Figure 2.5 would be:

$$TkknKKkknKKkkfjkcrKKkqknKFJKC$$

Because the implementation of CSSR requires the input data to contain alphanumeric characters, on top of the original serialisation, I perform the following additional data processing (the function $fromCharCode$ converts ASCII code to symbol):

- The original note on events $\{i \mid 1 \leq i \leq 128\}$ are mapped to the corresponding 12 pitch (on) classes represented by characters from "a" to "l".

  $$fromCharCode((i-1)\%12+97)$$

  For example, NOTE_ON_64 event (original index is 65) is denoted as

[1] `https://github.com/stites/CSSR`

"e".

- The original note-off events $\{i \mid 129 \leq i \leq 256\}$ are mapped to the corresponding 12 pitch (off) classes represented by characters from "A" to "L".

$$fromCharCode((i - 129)\%12 + 65)$$

For example, NOTE_OFF_64 event (original index is 193) is denoted as "E".

- The original time shift events $\{i \mid 257 \leq i \leq 356\}$ are mapped to 10 coarse-grained time shift events represented by characters from "m" to "v".

$$fromCharCode(\lfloor (i - 257)/10 \rfloor + 109)$$

For example, TIME_SHIFT_71 event (original index is 327) is denoted as "t".

- The original set velocity events $\{i \mid 357 \leq i \leq 388\}$ are mapped to 8 coarse-grained set velocity events represented by characters from "M" to "T".

$$fromCharCode(\lfloor (i - 357)/4 \rfloor + 77)$$

For example, SET_VELOCITY_20 event (original index is 376) is denoted as "Q".

Thus, there are a total of 42 unique symbols with the above process. CSSR requires the maximum history length, L, as a parameter of the pro-

gram. As suggested by the guideline:

$$L = \frac{log_2 N}{log_2 k}$$

where N is the length of sequence, that is fixed as 1024 in the case here, and k is the size of alphabet set, which is 42 as mentioned. So L is set as 1.855.

8.1.2 Translational complexity

Translational complexity is another measure used to detect the degree of randomness in structure, unlike statistical complexity, it requires the tested music to be the geometric representation described in Section 2.3.3. The applied translational complexity here is extended from the translational coefficient introduced by Foubert et al. (2017). The core procedure in calculating the translational complexity is to count the unique (start time, pitch)-differences between combinations of points. Suppose a set of points:

$$X = \{(0, 64), (1, 66), (3, 67), (4, 69)\}$$

from which 4 unique (start time, pitch)-difference pairs can be extracted:

$$\{(1, 2), (3, 3), (4, 5), (2, 1)\}$$

In an extremely simple case where the difference of each adjacent pair is at the same amount, there is a minimum $n - 1$ of pairs can be obtained,

where n is the size of the original set of points. For example:

$$X = \{(j + a, k + b), (2j + a, 2k + b), \cdots, (nj + a, nk + b)\}$$

where $\{j, k, a, b\} \in \mathbb{R}$, and the unique (start time, pitch)-difference pairs that can be extracted are:

$$\{(j, k), (2j, 2k), \cdots, ((n - 1)j, (n - 1)k)\}$$

which implies the difference between consecutive items of X is always (j, k); the difference between consecutive-but-one items is always $(2j, 2k)$; and so on; the difference between the first and last items is $((n - 1)j, (n - 1)k)$.

At the other extreme, it is possible to observe or define a maximally random set X of n points such that no two differences between pairs of items are the same, and in this case, there are $n(n-1)/2$ differences. For example:

$$X = \{(0, 64), (1, 62), (3, 67), (4, 69)\}$$

where 6 unique (start time, pitch)-difference pairs can be obtained:

$$\{(1, -2), (3, 3), (4, 5), (2, 5), (3, 7), (1, 2)\}$$

Thus, for any given set of n points, the theoretical minimum $(n - 1)$ and maximum $n(n - 1)/2$ pairs of (start time, pitch)-difference can be used to normalise the actual number of pairs m to $[0, 1]$. The translational complexity

is then formalised as:

$$trans\_comp = \frac{2(m-2+1)}{(n-1)(n-2)}$$

8.1.3 Arc score

The general idea of arc score is to conduct a multiple linear regression analysis to determine the melody contour, as represented as a set of (start time, pitch)-points. Section 7.5.3 reveals the correlation between the significance of melody contour and its ratings, an advanced score is expected here to quantify the significance. In most cases, the melody is located in the high-pitched area and the contour generally follows a bell curve. Based on this assumption, the melody data used to calculate arc score is extracted by simply picking up the notes with the highest pitch when multiple notes are played simultaneously. The extracted melody data is also in the geometric form but the future start time is strictly greater than the past:

$$X = \{(s_1, p_1), (s_2, p_2), \cdots (s_n, p_n) \mid s_n > s_{n-1}, n \in \mathbb{N}\}$$

where s, p and n respectively denote the start time, pitch and size of the set. To detect whether the melody contour is a bell curve, a second-degree polynomial can be used to fit to the data, with two hypotheses about the coefficient β:

- $H_0 : \beta = 0$, which implies no statistically significant relationship between the predictor variable S (the start time set) and the response variable P (the pitch set).

- $H_1 : \beta \neq 0$, which implies, on the contrary, there is significant relationship between S and P.

To obtain the estimated coefficients $\hat{\beta}$, the polynomial linear regression is implemented by a JavaScript package named "ml-regression-polynomial", which fits the polynomial to the above-mentioned melody data and returns the estimated coefficients.[2] Afterwards, a t-test is performed on the estimated second-degree coefficient to check its significance, which is the arc score defined here. As only the second degree is considered here, $\hat{\beta}$ and β will only indicate the (estimated) second-degree coefficient from now on. The test statistic is the ratio of the difference between the statistic and the parameter and the standard error of the statistic, and it is based on the t-distribution:

$$\frac{\hat{\beta} - \beta}{\sqrt{\frac{RSS/(n-k-1)}{SS}}} \sim t(n-k-1)$$

where $\hat{\beta}$ is the estimated coefficient, β is zero corresponding to H_0, k is the number of predictors and $t(n-k-1)$ denotes the t-distribution with $(n-k-1)$ degrees of freedom, k is 2 in this case, RSS is the residual sum of squares:

$$RSS = \sum_{i=1}^{n} (p_i - f(s_i))^2$$

where $f(s_i)$ is the estimated pitch value with the given start time value, and SS denotes the sum of squares:

$$SS = \sum_{i=1}^{n} (s_i - \bar{s})^2$$

[2] `https://github.com/mljs/regression-polynomial`

where $\bar{s}$ is the mean value of the start time set S.

Once the test statistic is obtained, the final step is to acquire the p-value or arc score, which describes the probability of obtaining test results at least as extreme as the actually observed results, under the assumption of H_0 being correct. This is done by calculating the cumulative distribution function with the above test statistic for the t-distribution with $(n-k-1)$ degrees of freedom.

8.1.4 Tonal ambiguity

Krumhansl (2001) describes the Krumhansl-Schmuckler key-finding algorithm where the distribution of pitch classes in a piece is compared with key profiles for each key, it requires the input data to have pitch and duration information and gives the correlation coefficient to a certain key as return. This key-finding algorithm is based on predefined major and minor key profiles as shown in Table 8.1, each profile consists of 12 values corresponding to pitch classes.

C Major profile											
C	C♯	D	D♯	E	F	F♯	G	G♯	A	A♯	B
6.35	2.23	3.48	2.33	4.38	4.09	2.52	5.19	2.39	3.66	2.29	2.88
C Minor profile											
A	A♯	B	C	C♯	D	D♯	E	F	F♯	G	G♯
6.33	2.68	3.52	5.38	2.60	3.53	2.54	4.75	3.98	2.69	3.34	3.17

Table 8.1: C major and C minor key profiles.

Here, only C major and C minor key profiles are demonstrated, the other key profiles can be obtained by just rolling the values. As shown in Table 8.2,

shifting values by two places to the right gives the key profile of D major

D Major profile											
C	C♯	D	D♯	E	F	F♯	G	G♯	A	A♯	B
2.29	2.88	6.35	2.23	3.48	2.33	4.38	4.09	2.52	5.19	2.39	3.66

Table 8.2: D major key profile.

The values of the profile are obtained from the listening experiment conducted by Krumhansl (2001). The participants of the experiments are requested to listen to a set of context tones and a probe tone, and then asked to rate the extent to which the probe tone fits with the context. The sum of durations for 12 pitch classes needs to be calculated for the piece to be compared. For example, considering a set of points consisting of 4 notes, each of which is represented as a (start time, pitch, duration)-tuple:

$$\{(0, 60, 2), (0, 72, 2), (1, 64, 1), (1, 67, 1)\}$$

where the duration sum for each pitch class is shown in Table 8.3.

C	C♯	D	D♯	E	F	F♯	G	G♯	A	A♯	B
4	0	0	0	1	0	0	1	0	0	0	0

Table 8.3: An example duration sum.

The correlation coefficient R between a key profile and the duration sum for a piece can be obtained by:

$$R = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

where the values of a key profile and duration sum can be substituted as x and

y, the value of R is in the range of $[-1, 1]$: 0 indicates no linear relationship; 1 indicates a positive correlation; -1 indicates a negative correlation. With a given piece, the algorithm calculates R for each of the 12 major and minor key profiles, and returns the one with the largest value. For the example in Table 8.3, the algorithm assigns C major as the most correlated key with $R = 0.831$.

8.1.5 Time intervals

Rhythmic features can be obtained from various types of time intervals. Bresin and Umberto Battel (2000) provides a statistical analysis of pianists' expressive performances by investigating inter-onset-interval (IOI), key overlay time (KOT) and key detached time (KDT). Figure 8.2 defines these three intervals.

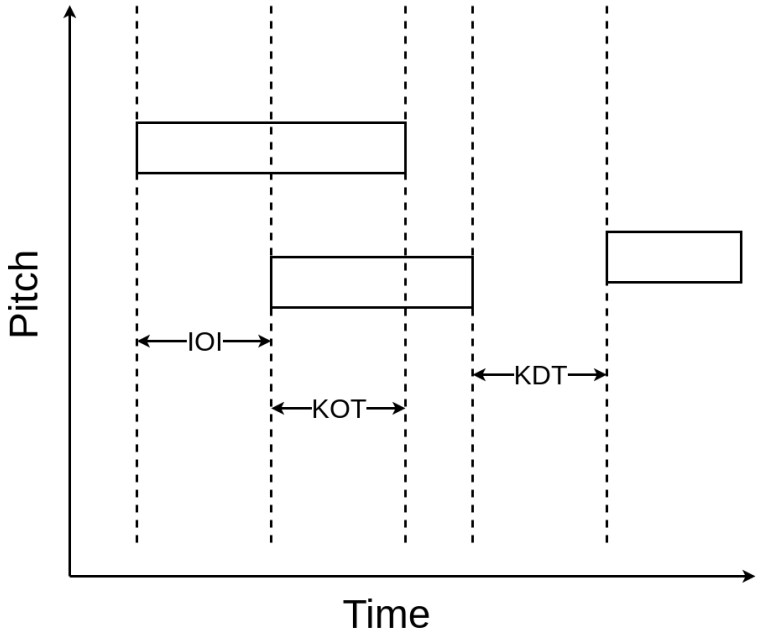


Figure 8.2: Notes are represented by bars, where IOI is the time difference between the onsets of adjacent notes; KOT is the time difference between the offset of a note and onset of the successive overlapped note; KDT is the time difference between offset of a note and onset of the successive non-overlapped note.

For a given piece, three sets of IOIs, KOTs and KDTs can be extracted, where the IOI set size equals the sum of KOT and KDT set sizes, because two adjacent notes can only be either overlapped or non-overlapped. The mean of an interval set can be calculated, it represents the average difference between times. However, merely using the mean is insufficient to capture the expected rhythmic characteristics, which can be demonstrated with the following example of IOI sets with the same mean:

$$\mathrm{IOI}_A = \{4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4\}$$

$$\mathrm{IOI}_B = \{2, 5, 3, 6, 2, 5, 3, 6, 2, 5, 3, 6\}$$

where IOI_A implies a simple rhythmic pattern where every note is played

after a constant gap, but IOI_B implies a more complex rhythm. According to this, the variance needs to be also considered. In general, it measures the randomness of rhythmic patterns, a low variance indicates a steady rhythm, whereas a high variance indicates that the time differences are more diverse.

There are more subtle rhythmic differences which cannot be distinguished by the above method. Considering the following example:

$$\text{IOI}_B = \{2, 5, 3, 6, 2, 5, 3, 6, 2, 5, 3, 6\}$$
$$\text{IOI}_C = \{2, 2, 2, 3, 3, 3, 5, 5, 5, 6, 6, 6\}$$

where the mean and variance of IOI_B and IOI_C are the same, the rhythmic pattern is totally different. Thus, the second-level IOI is introduced to solve the issue, it is simply calculated by subtracting two consecutive (first-level) IOIs. For example, the second-level IOI_B and IOI_C is:

$$\text{IOI}_B^{\text{II}} = \{3, -2, 3, -4, 3, -2, 3, -4, 3, -2, 3\}$$
$$\text{IOI}_C^{\text{II}} = \{0, 0, 1, 0, 0, 2, 0, 0, 1, 0, 0\}$$

where the mean and variance of IOI_B^{II} and IOI_C^{II} are different.

8.1.6 Onset jitters

Music performers often add expressive timing and dynamics to the original notations, so that the same piece can be expressed differently. As mentioned, Bresin and Umberto Battel (2000) investigate the effects of time intervals on expressive performance, and various statistics of time intervals are introduced in this study to measure the rhythmic features. Section 7.5.3 demonstrates

that models can generate music with expressive timing, although the generated time variations occasionally make the whole music drop from the beat. Figure 8.3 shows two types of onset jitter which can be made to a note.

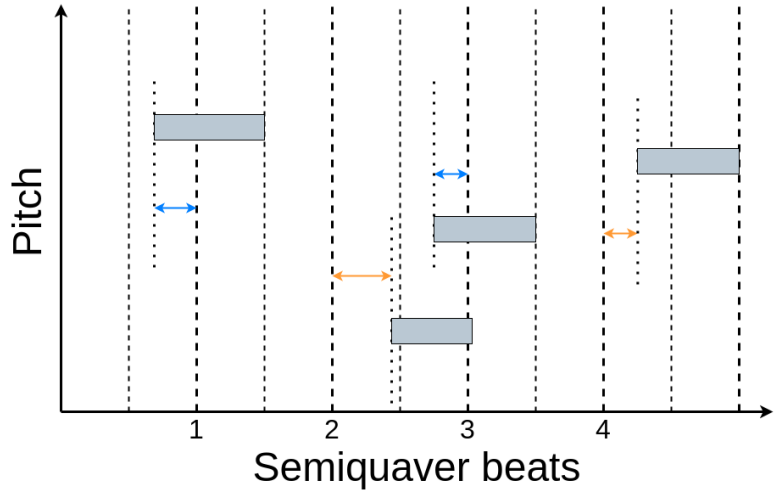


Figure 8.3: Notes are represented by bars, the time indicated by blue lines is the amount of onset jitter that rushed to the nearest semiquaver beat, and the time indicated by orange lines is the amount of onset jitter that delayed after the nearest semiquaver beat.

Similar to the statistical analysis of time intervals, with a given set of note points, the jitters of rush and delay are separately collected, and the mean and variance are calculated to measure the expressiveness of onsets. The mean is further normalised into the range of $[0, 1]$ by dividing the duration of a demisemiquaver, as it is the maximum amount of jitter that can be made. A high jitter mean indicates that the onsets of notes are mostly dropped from the beat, and a high mean with a low variance can imply that the beat is shifted by a certain amount of time.

As described in Chapter 4, Yang and Lerch (2020) introduces a set of pitch- and rhythm-based features to characterize a music corpus. For comparison, the features I present here include not only pitch and rhythm-based fea-

tures, but also statistical complexity, translation complexity and arc scores, which are specifically designed to measure structural information musicology.

8.2 Results

To demonstrate the effectiveness of above features, I calculate the feature values for both model-generated and human-composed excerpts mentioned in Section 7.4.2. Based on the collected ratings in six musical dimensions, I then make raincloud plots (Allen et al. 2019) to show the distributions of ratings and feature values. Due to the limited space, I only include plots that are relevant to key findings. However, the whole set of plots can be accessed in Appendix B.

As the raincloud plots here contain various elements with the same format, I first use Figure 8.4 to provide a general introduction.

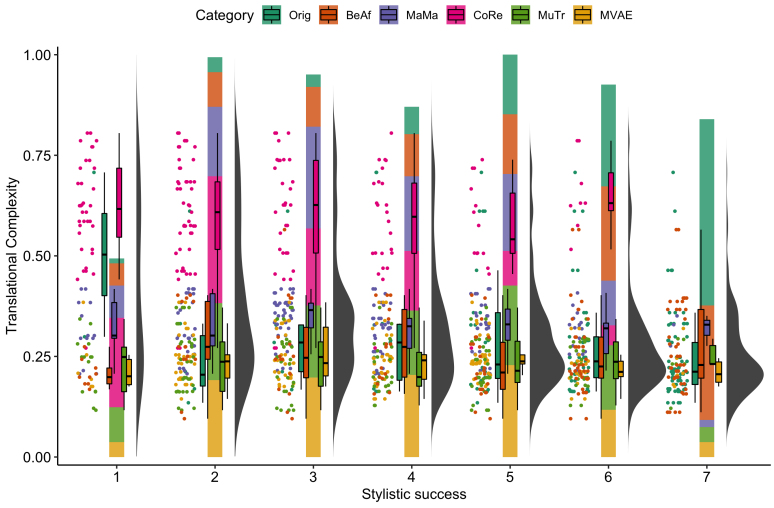


Figure 8.4: Translational complexity against stylistic success ratings for CSQ excerpts.

The x-axis indicates the stylistic success rating from 1 to 7, and the y-axis

indicates the translational complexity in the range of $[0, 1]$. As shown at the top, I use different colours to denote categories of excerpts in CSQ part of study. For each individual rating, there are:

- scatter points denoting the presence of excerpts;
- box plots respectively showing the minimum, the lower quartile, the median, the upper quartile and the maximum of values;
- a bar plot with segments denoting the ratio of numbers of categories;
- a flat violin plot[3] denoting the continuous distributions of those scatter points.

8.2.1 Statistical complexity

The calculation of statistical complexity requires a batch of excerpts instead of a single excerpt for other features. Thus, only the plot for statistical complexity is different from the remaining features in this section. I show statistical complexity for each category in Figure 8.5, which is developed based on Figure 7.2 in Section 7.5, the violin plot for each category is colored to represent statistical complexity.

[3]The flat version removes the mirrored half of the whole violin.

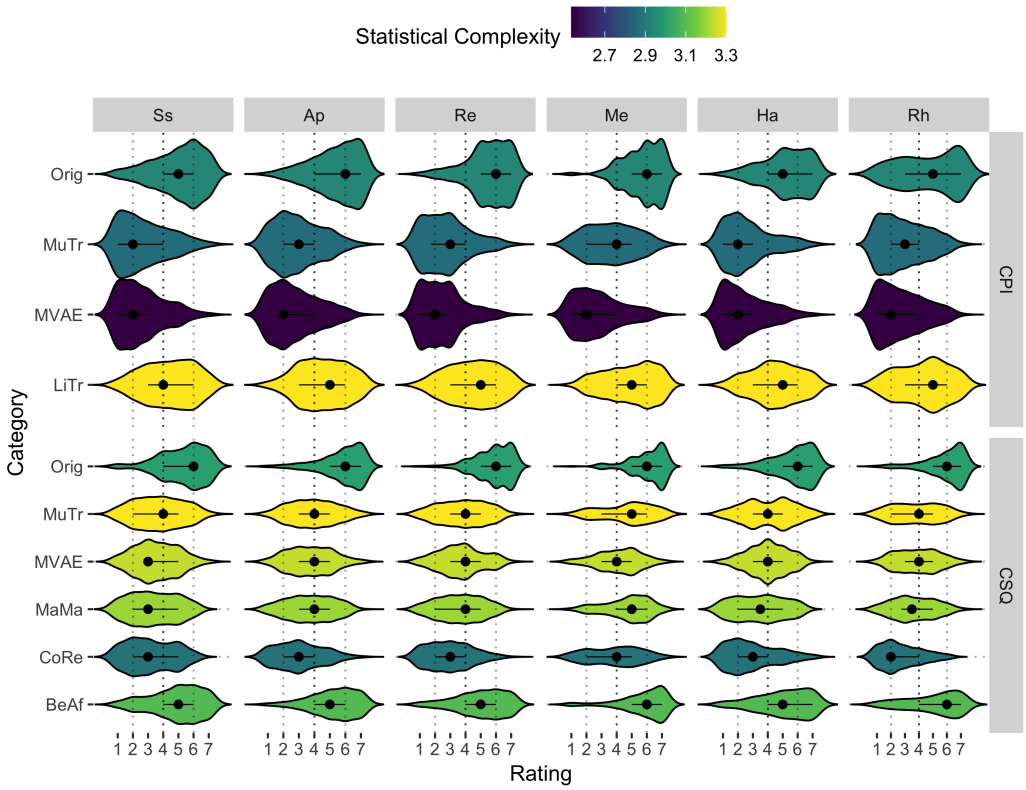


Figure 8.5: Rating (1–7) distributions of six musical dimensions with colour filled to represent statistical complexity

The exact values are also shown in Table 8.4. *Orig* in both CSQ and CPI, and *BeAf* in CSQ have a very close statistical complexity of around 3, and also these categories receive the highest ratings on every musical dimension. Considering the complexity of human-composed music as a baseline, it can be seen that other generated categories have either higher or lower statistical complexity, and receive lower ratings. This fact of statistical complexity and ratings demonstrate the concept shown in Figure 8.1, that is, music with generally good quality (human-composed) remains at a certain level of complexity, any excerpt above or below this level is very likely to have lower quality. Although, *CoRe* is an exception, which has statistical complexity of

2.878 but almost the lowest overall ratings.

Table 8.4: Statistical complexity for each category.

	Classical string quartet (CSQ)	Classical piano improvisation (CPI)
Original (Orig)	3.004	2.937
Before-After (BeAf)	3.090	–
MAIA Markov (MaMa) (Collins and Laney 2017; Collins et al. 2016b)	3.198	–
Coupled Recurrent Model (CoRe) (Thickstun et al. 2019)	2.878	–
MusicVAE (MVAE) (Roberts et al. 2018)	3.240	2.534
Music Transformer (MuTr) (Huang et al. 2018)	3.298	2.864
Listen to Transformer (LiTr) (Huang et al. 2018)	–	3.300

8.2.2 Translational complexity

Regarding translation complexity in CSQ part, Figures 8.4 and 8.6 show higher kurtosis of distribution for higher ratings (5, 6, 7), but the plots of other four musical dimensions do not have such significant differences of ratings (see Appendix B). It implies that the excerpts with translational complexity around 0.2 –20% unique pairs of (start time, pitch) difference– are more likely to receive higher ratings of stylistic success and repetition. In opposite, the majority excerpts of *CoRe* have high translational complexity, leading to poor performance as shown in Section 7.5.

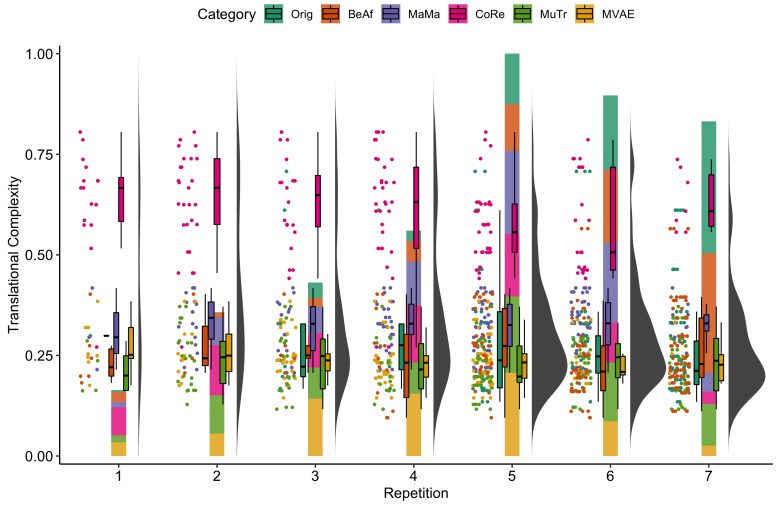


Figure 8.6: Translational complexity against repetition ratings for CSQ excerpts.

Whereas in CPI, Figure 8.7 shows the opposite relation between translational complexity and stylistic success, where excerpts with higher translational complexity are more likely to receive a lower stylistic success rating. Such a difference to CSQ is because the expressive timing of CPI excerpts increases the amount of unique (start time, pitch) pairs.

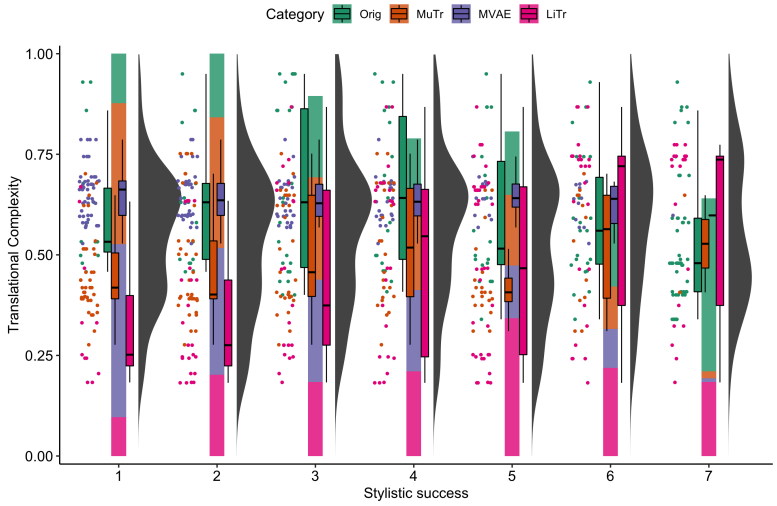


Figure 8.7: Translational complexity against stylistic success ratings for CPI excerpts.

For repetition ratings in CPI, Figure 8.8 does not show the same pattern as CSQ. As the overall repetition rating for CPI is lower than CSQ, it is more challenging for AMG systems to learn strong repetitive structures of piano improvisations than string quartets.

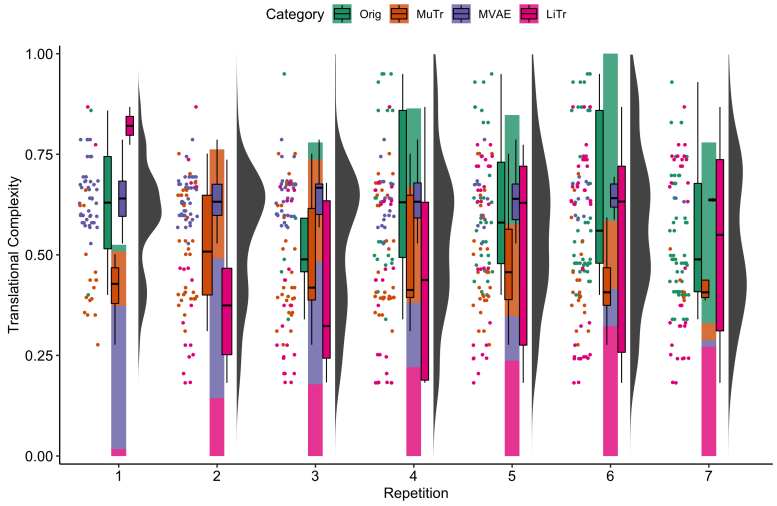


Figure 8.8: Translational complexity against repetition ratings for CPI excerpts.

8.2.3 Arc score

By examining the plots in Appendix B.1.2 and B.2.2, Arc score does not seems to have a specific pattern for any of these musical dimensions and styles.

However, based on the musicological analyses demonstrated in Section 7.5.3, where I mentioned the arc shape of melody, here I again select some of them with different arc shapes as shown in Figure 8.9, and measure their arc scores:

a with an up and down arc shape has an arc score of 0.796;

b with a down and up arc shape has an arc score of 0.693;

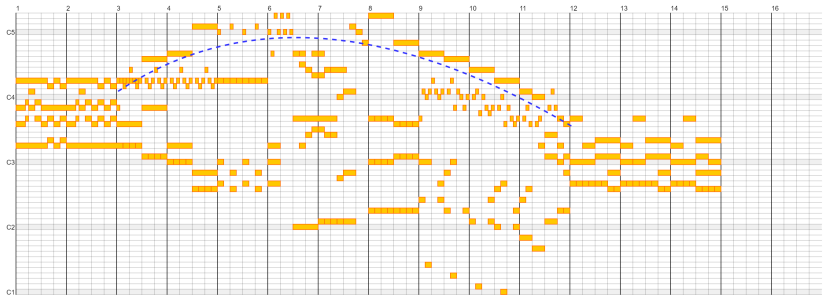
c with a rising arc shape has an arc score of 0.643;

d with a nearly flat arc shape has an arc score of 0.501.

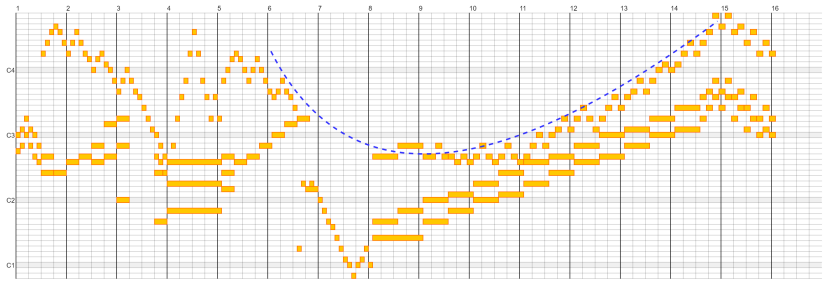
Comparing the arc shapes and their corresponding scores demonstrates the capability of arc scores measuring the significance of arc shapes of melody, for both directions (see Figure 8.9 (a) and (b)). Although arc score does not show a significant correlation to ratings, I reason that the arc shape is not a dominant factor for any musical dimension, it still measures the expected properties.

8.2.4 Tonal ambiguity

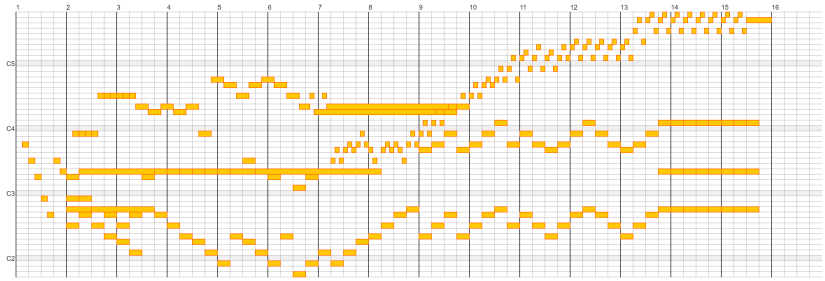
Generally, CSQ excerpts with lower tonal ambiguity receive higher ratings, especially for stylistic success and melody as demonstrated in Figures 8.10 and 8.11.



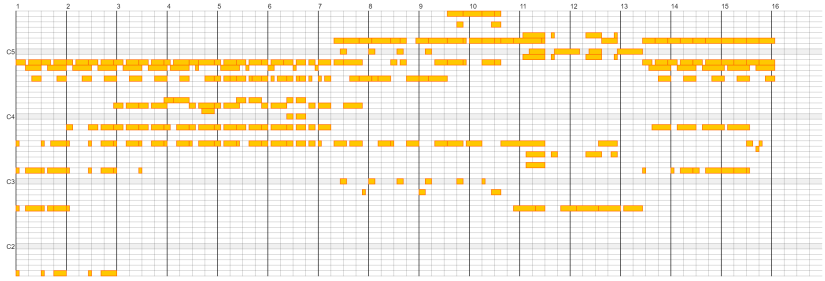
(a)



(b)



(c)



(d)

Figure 8.9: Excerpts with different melody arc shapes, where (a) and (b) show clear melody arcs in opposite directions, (c) shows a rising melody and (d) shows a nearly flat melody line.

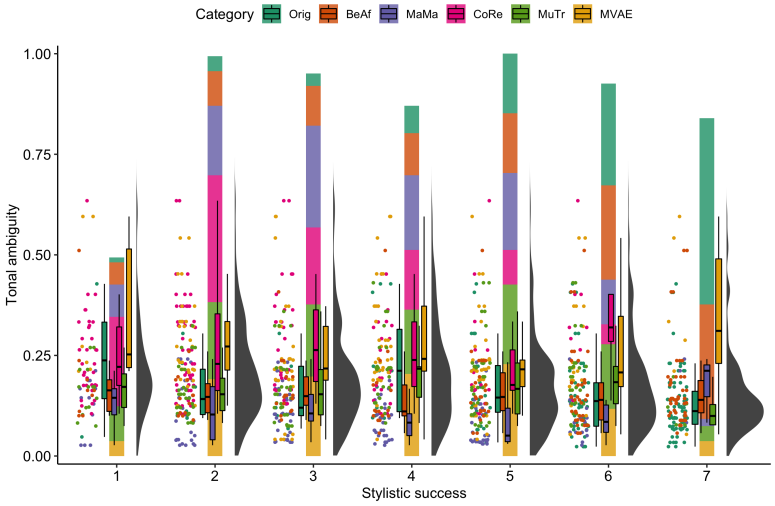


Figure 8.10: Tonal ambiguity against stylistic success ratings for CSQ excerpts.

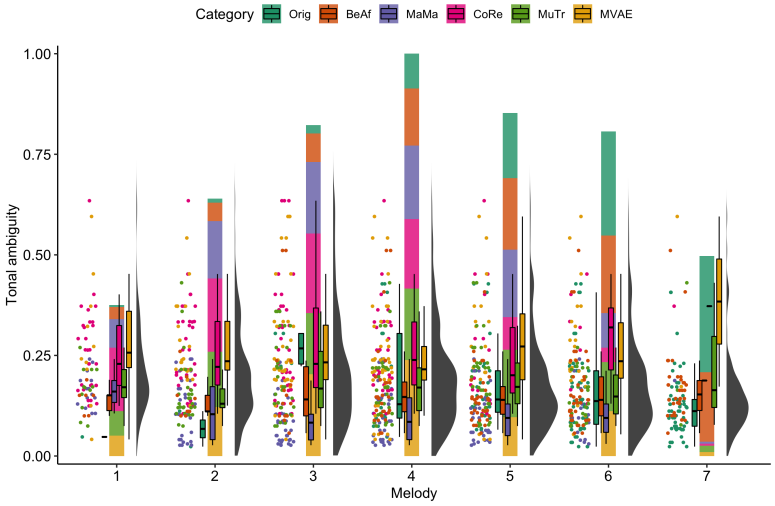


Figure 8.11: Tonal ambiguity against melody ratings for CSQ excerpts.

CPI excerpts have similar patterns as CSQ, but the slope of dropping ambiguity is steeper than CSQ, as shown in Figure 8.12 and 8.13 for melody and harmony ratings.

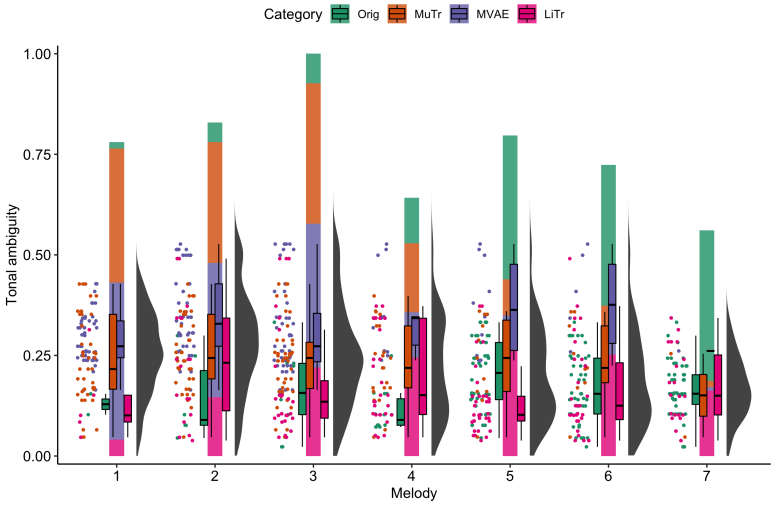


Figure 8.12: Tonal ambiguity against melody ratings for CPI excerpts.

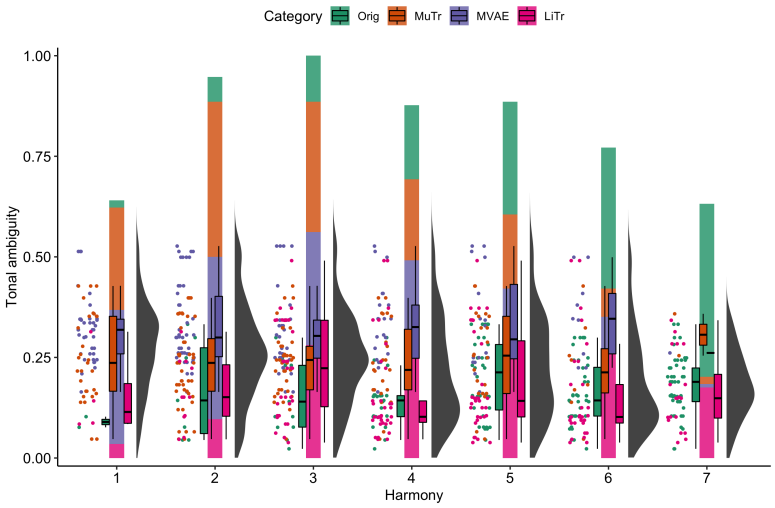


Figure 8.13: Tonal ambiguity against harmony ratings for CPI excerpts.

According to the above figures, tonal ambiguity is strongly correlated with consonance/dissonance of excerpts, mainly represented by melody and harmony ratings. The steeper slope in the CPI part demonstrates that tonal ambiguity varies depending on styles.

8.2.5 Time intervals

Intuitively, time interval features are supposed to be relevant to rhythm ratings. In the CSQ part, Figure 8.15 and 8.14 show rhythm and repetition ratings from low to high with a clear decreasing kurtosis of KOT mean.

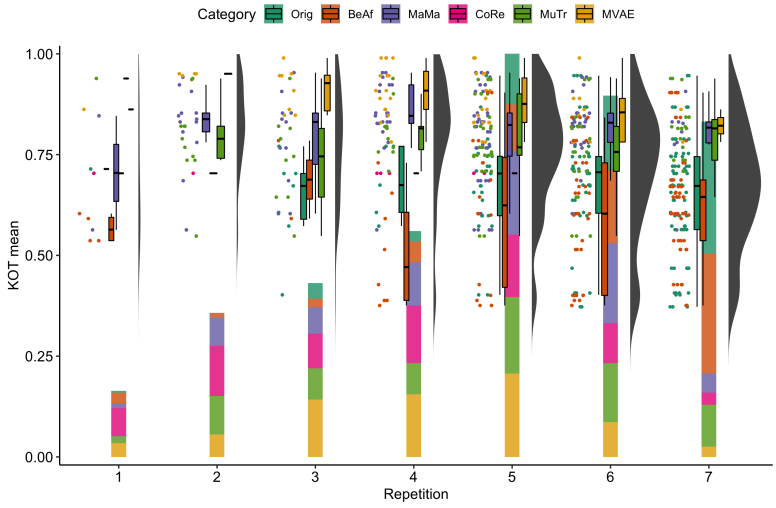


Figure 8.14: KOT mean against repetition ratings for CSQ excerpts.

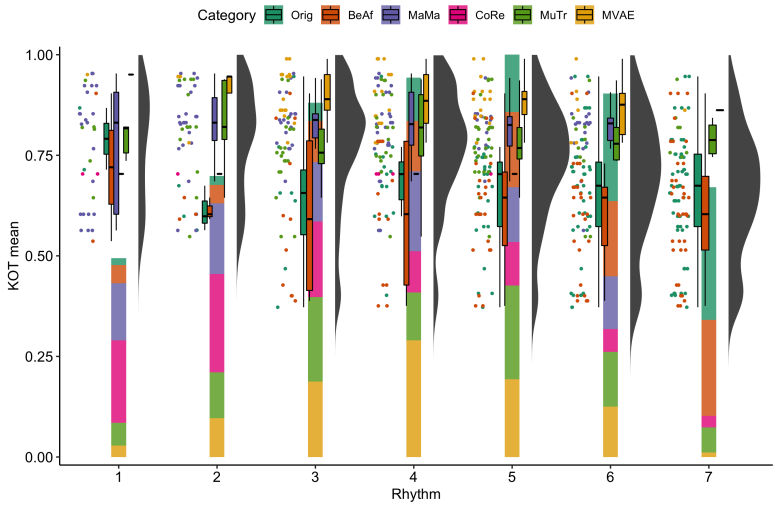


Figure 8.15: KOT mean against rhythm ratings for CSQ excerpts.

And also, Figure 8.16 demonstrates that excerpts with higher rhythm ratings tend to have lower KOT variance.

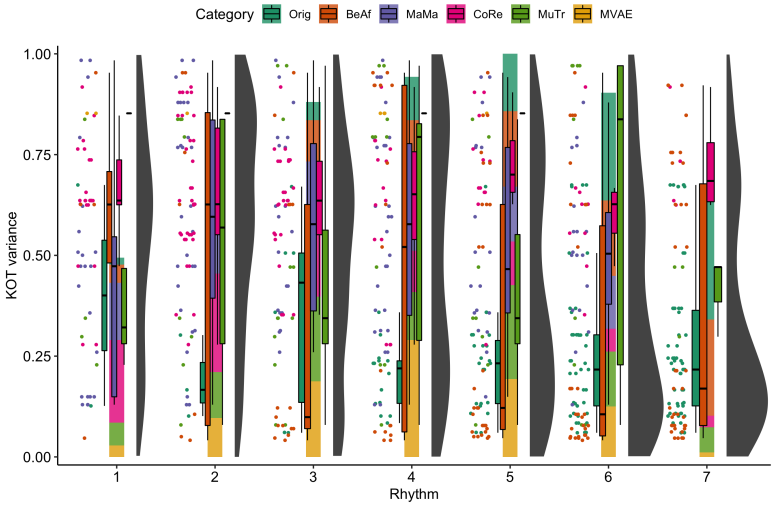


Figure 8.16: KOT variance against rhythm ratings for CSQ excerpts.

For other features, such as IOI and IOI$^{\text{II}}$ shown in Figure 8.17 and 8.18, the variance of excerpts with higher ratings is restricted to a narrower range.

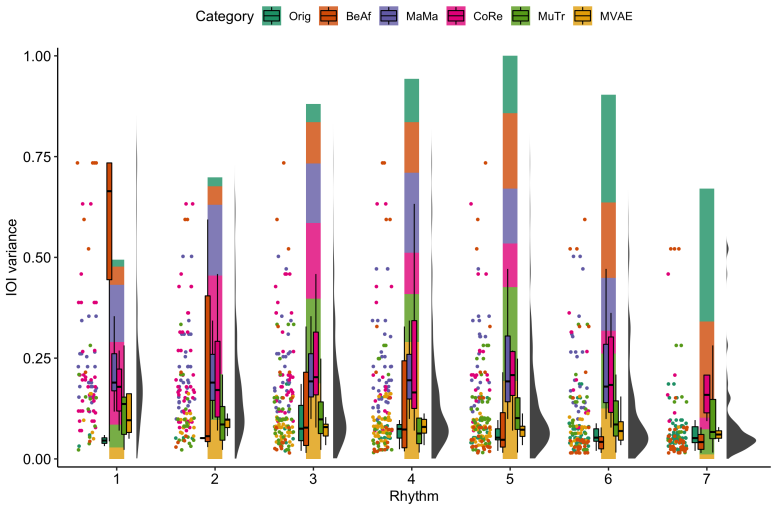


Figure 8.17: IOI variance against rhythm ratings for CSQ excerpts.

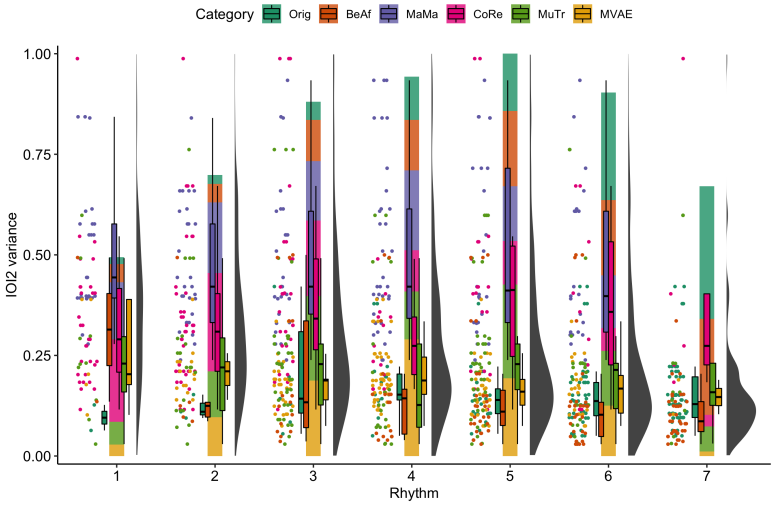


Figure 8.18: IOI^{II} variance against rhythm ratings for CSQ excerpts.

However, the CPI part does not show the same correlation as CSQ, as expressive timing adds more noise when calculating these time interval features. Figure 8.19 implies that, at approximate 0.2 IOI, it is more likely that an excerpt receives lower rhythm ratings.

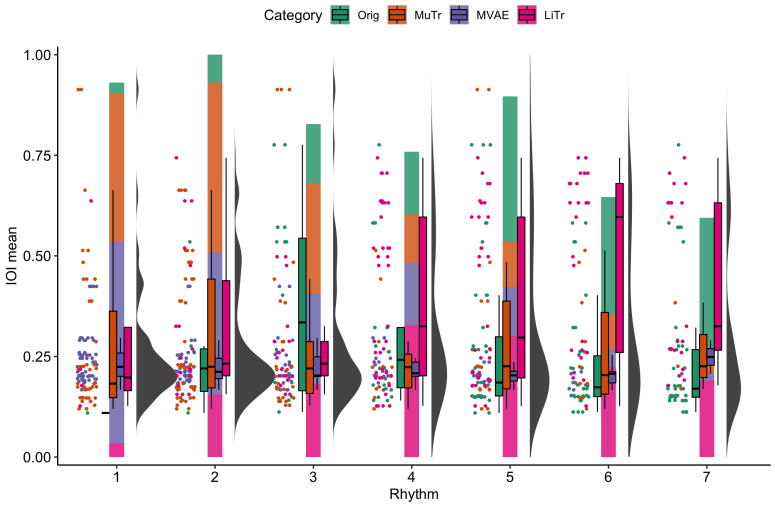


Figure 8.19: IOI mean against rhythm ratings for CPI excerpts.

Figure 8.20 shows KDT variance dropping to zero from low to high

rhythm ratings.

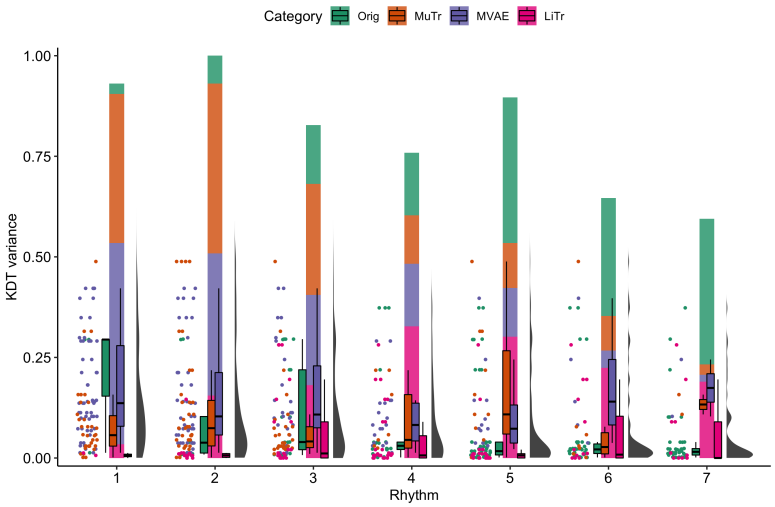


Figure 8.20: KDT variance against rhythm ratings for CPI excerpts.

8.2.6 Onset jitters

Onset jitters measure the overall subtle variations of notes start time. In the CSQ part of study, Figures 8.21 and 8.22 show that excerpts with higher ratings tend to have nearly zero jitters.

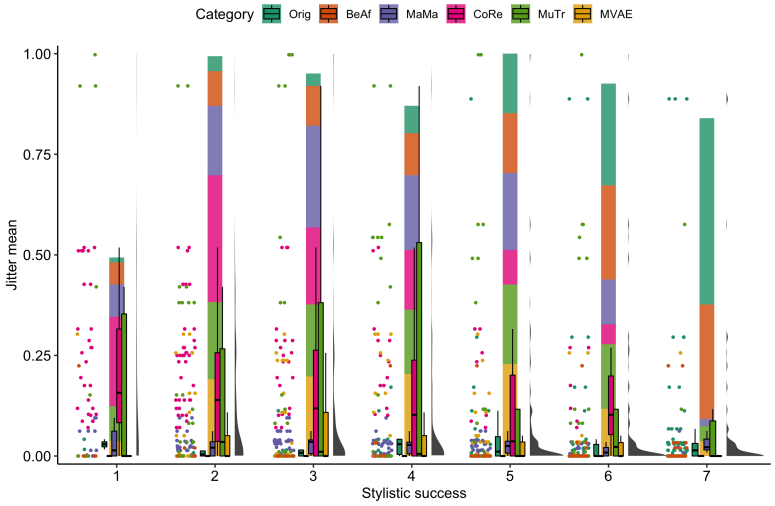


Figure 8.21: Onset jitters mean against stylistic success ratings for CSQ excerpts.

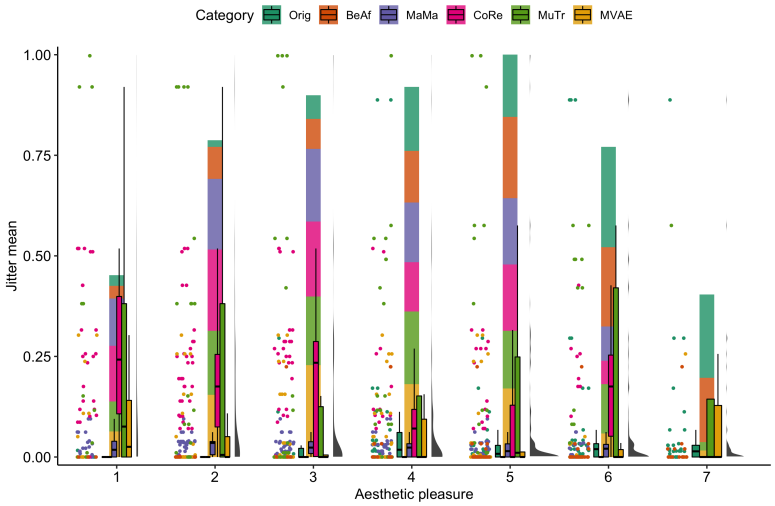


Figure 8.22: Onset jitters mean against stylistic success ratings for CSQ excerpts.

On the contrary, in the CPI part of study, Figures 8.23 and 8.24 indicate that the excerpts with higher ratings tend to have a wider range of jitters, because piano improvisations are expected to have expressive timing, which should contribute to stylistic success and aesthetic pleasure. However, excerpts in *LiTr* show zero onset jitters, which means the start time of each

note is on the semiquaver grid.

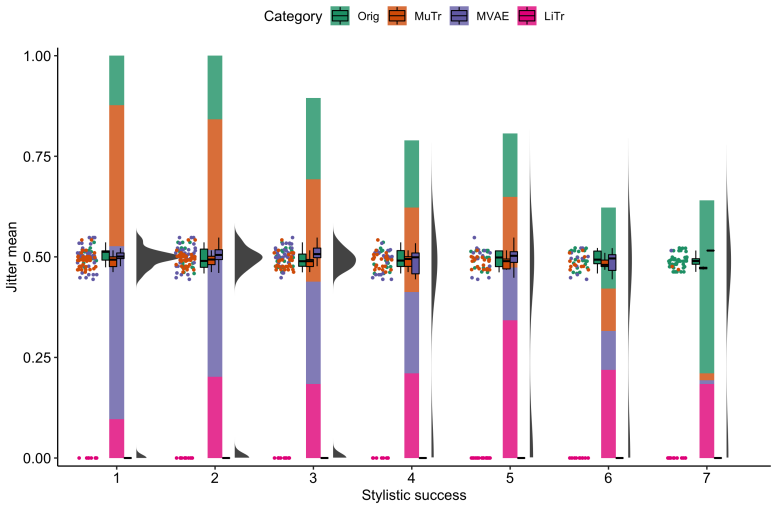


Figure 8.23: Onset jitters mean against stylistic success ratings for CPI excerpts.

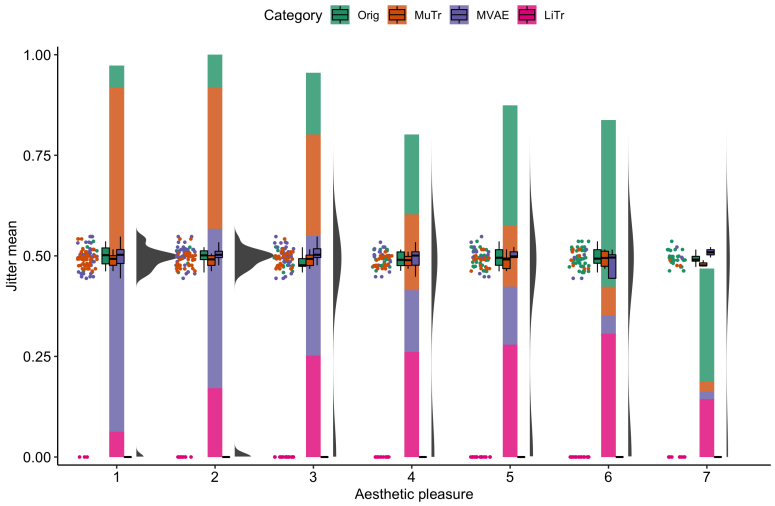


Figure 8.24: Onset jitters mean against stylistic success ratings for CPI excerpts.

Table 8.5: Effectiveness of features for modeling musical dimensions

	Ss	*Ap*	*Re*	*Me*	*Ha*	*Rh*
Translational complexity	✓		✓			
Arc score						
Tonal ambiguity				✓	✓	
Time intervals			✓			✓
Onset jitters	✓					

8.3 Discussion

This chapter addresses research question 3, I introduce six musical features, and analyse their effectiveness in measuring musical dimensions, by investigating their correlation to ratings.

AMG systems often use different model architecture and generation strategy, it is difficult to find a systematic way to compare their performance. Here I conduct statistical analyses on the extracted features and musical dimensions introduced in Chapter 7, Table 8.5 summarise the effectiveness of each feature, statistical complexity is omitted because its calculation requires a whole set of pieces, unlike other features that can be extracted from each piece, but statistical complexity is analysed in Figure 8.5.

Statistical complexity describes the minimum amount of historical information required to make optimal predictions, applying such measure enables a direct comparison between generative systems (including both human composers and generative models). The results show that human-composed excerpts share the same level of statistical complexity, while excerpts generated by systems have either higher or lower statistical complexity and receive lower ratings in all musical dimensions than human-composed excerpts. However,

statistical complexity does not absolutely predict the overall performance, *CoRe* have very close statistical complexity to *Orig* but also lowest ratings. As shown by translational complexity, the significance of the relation between features and musical dimensions can vary depending on styles. In CSQ, it shows the excerpts with higher repetition ratings tend to have lower translational complexity, whereas, CPI excerpts with lower stylistic success ratings tend to have higher translational complexity. As the current version of translational complexity is sensitive to expressive timing, it makes the results of the two styles not comparable. Example excerpts (see Figure 8.9) demonstrate that arc score is able to measure the significance of melody arc shape. However, arc score does not show an apparent correlation to musical dimensions, I infer the melody arc is not a dominant factor when participants perceive the excerpts and rate in the aspect of these musical dimensions. Tonal ambiguity shows more straightforward patterns, the melody and harmony increase when tonal ambiguity decreases. The CPI part of study shows a steeper decrease than the CSQ part. Regarding time interval features, the results show that with higher repetition and rhythm ratings, KOT mean decreases in the CSQ part of study, and KOT variance also has a negative linear correlation to rhythm ratings. The higher-rated excerpts have lower IOI and IOI^{II} variance. Like translational complexity, onset jitters measure differently with styles. As CSQ excerpts have no expressive timing, the excerpts with higher stylistic success ratings tend to have onset jitters near zero. Although CPI is supposed to have expressive timing naturally, the results do not show an exactly opposite pattern to CSQ, it is because the current method of onset jitters measuring does not distinguish the goodness/badness of the generated

jitters. Thus, this feature needs further improvement to handle expressive timing properly.

This chapter focuses on the correlation of proposed musical features and ratings, and explores the possibility of modelling human judgment through combinations of these features. This concludes the main body of this thesis, the next chapter summarises the three technical chapters presented to address the three research questions proposed in Chapter 5. Finally, I reiterate the contribution of this thesis and suggest possible future work.

9

Conclusions and future work

Conclusions of technical Chapters 6, 7 and 8 are made respectively in Sections 6.6, 7.6 and 8.3. In this ending chapter, I refine the key findings and contributions to answer the research questions listed in Chapter 5 and suggest future work that can be done to improve current methods and contribute to the field.

In the literature review, I first provide an introduction to music representations in Chapter 2, which is a necessary preliminary for understanding how the music notation can be translated into various data formats and enable the development of AMG systems. In Chapter 3, I then review representative symbolic AMG works in categories: rule-based approaches, sequential models, artificial neural networks, and deep learning approaches. To elicit the main questions in this field, I discuss existing works evaluating generic creative systems and AMG systems in Chapter 4. I also illustrate the "echo chamber" of deep learning-based AMG literature and non-trivial issues of their evaluation methods, including the lack of 1) approaches for analysing the originality of generated samples; 2) and a set of standard and comprehensive metrics in musical dimensions. Particularly, some deep learning-based AMG works tend to evaluate generation performance based on machine learning losses, which makes it impossible to compare deep learning-based systems

with different loss functions, and systems that are not based on deep learning methods (e.g., MAIA Markov Collins and Laney 2017). Moreover, machine learning losses do not necessarily reflect the generation quality in musical dimensions, and musicological analysis of the generated music is rarely demonstrated in these works.

To what extent the current models generate original music compared to human composers baseline?

To address research question 1, I present an evaluation framework in Chapter 6, to measure the extent to which a generative model copies from its training data. This approach is parameterised by a similarity measure, and I demonstrate it with cardinality score (Ukkonen et al. 2003; Collins et al. 2014) and fingerprinting score (Arzt et al. 2012; Collins et al. 2016a; Wang and Smith III 2012). The former uses a more straightforward note-counting concept to enable fast computation, but it cannot adapt to expressive timing, while the latter handles expressive timing well, it requires more computational time and space. I apply both similarity measures to analyse outputs from Music Transformer (Huang et al. 2018) and MAIA Markov (Collins and Laney 2017), the results demonstrate the use of this evaluation methodology and the music plagiarism issue of recent research. The key findings of this part of work are as follows:

1. The originality of Transfomer's output is below the 95% confidence interval of the baseline.

2. The Transformer model obtained via the conventional stopping criteria

produces single-note repetition patterns, resulting in outputs of low quality and originality.

3. While in later training epochs, the model tends to overfit, producing copies of excerpts of input pieces.

4. Even with a larger dataset, the above issues still exist.

This work contributes to the field with an evaluation framework for measuring the originality of generated music. The measured originality is relative to the baseline, not absolute, to reflect the varying degrees of variety that exist between music styles. As this framework is parameterised by a similarity measure, designing other measures allows adapting the methodology to have emphases on different musical dimensions and handle data with different characteristics. The main limitation of the presented method is the lack of consideration of distinctiveness, meaning it cannot distinguish between the commonly used compositions within the corpus and distinctive compositions of individual pieces, where the latter should be more focused. According to the findings and contributions above, I list the following future work:

1. Future research on AMG should evaluate the originality of the generated outputs, so as to judge whether the model is learning to be a "memoriser" (stealing) or a "creator" (imitating) during the training process.

2. Specifically for AMG with deep learning, future work can investigate the feasibility of embedding the originality report method into training processes (e.g., as a loss function), thus the model can be pun-

ished/awarded by stealing/imitating the training data, and the generated outputs can be naturally more original.

3. To improve the current method, weighting the fingerprints based on their distribution in a dataset potentially gives more attention to distinctive parts and ignores common parts (e.g., consecutive single notes).

In musicological aspect, what is the quality gap between deep and non-deep learning music generation systems, human composers and these AI-based generation systems?

Deep learning methods may be improving relative to one another according to evaluations of losses, but these losses appear to be of limited use. Ultimately, it is not known how such improvement in losses is reflected in listening experience. To interpret the performance of AMG systems in various musical dimensions, and find their quality gap with human composers, I conduct a comparative evaluation in Chapter 7, which includes four AMG systems: a non-deep learning model (Collins and Laney 2017), and three deep learning models with different generation strategies (Huang et al. 2018; Roberts et al. 2018; Thickstun et al. 2019). I train these models with datasets in two target styles: Classical string quartets and classical piano improvisations. Nine hypotheses about comparing systems' performance are drawn up (see Section 7.1). Mixed with human-composed and generated excerpts, I then conduct a listening study where the participants rate the excerpts generated by AMG systems in six musical dimensions: stylistic success, aesthetic

pleasure, repetitive structure, melody, harmony, rhythm. I finally apply the non-parametric Bayesian hypothesis testing to the ratings to verify my hypotheses (see results in Section 7.5.2). Here I list the key findings as answers to research question 2:

1. A large gap still exists between the stylistic success ratings for the strongest-performing AMG systems and the human-composed excerpts; and none of these AMG systems achieves the same level as human-composed excerpts in any rating.

2. The assumption of superiority of deep learning methods for AMG is unwarranted; a deep learning method, Music Transformer (Huang et al. 2018), has equivalent performance in stylistic success to a non-deep learning method, MAIA Markov (Collins and Laney 2017), which demonstrates that deep learning, to date, does not outperform other methods for AMG.

3. With the "Carol of the Bells" example mentioned, Music Transformer is again shown to exceed the level of borrowing considered reasonable among human composers.

The research effort devoted to evaluation and evaluation methodologies has fallen behind the efforts invested in training and generating with novel network architectures (Pearce and Wiggins 2001; Jordanous 2012; Yang and Lerch 2020). There had been no such comparative evaluation existed for AMG before this work. Regarding the contributions here, I fill the gap in comparative evaluation of AMG systems, between deep and non-deep learning-based methods. In particular, I demonstrate the extent to which

the recent AMG system improve in terms of listening experience, which is reflected by participants' ratings in various musical dimensions. With the ratings collected, my musicological analyses of generated excerpts shed some light on the origin of the gap with human-composed excerpts (see Section 7.5.3). Furthermore, the entire framework can serve as a reference for future comparative evaluation work, especially the use of Bayesian methods (van Doorn et al. 2020) across the evaluation of machine learning systems. According to the findings and contributions described, I propose the following future work:

1. The term "style" in the context of AMG can refers to two separated concepts: musical style (e.g., Conklin and Witten 1995; Collins et al. 2016b) and performance style (e.g., Widmer 2002; Grachten and Widmer 2011). The six music dimensions introduced in my listening study are mostly relevant to musical style. As there are also other generative models for performance style. This evaluation framework can be improved by investigating the performance-specific musical dimensions.

2. Pieces are often simply sliced for training AMG systems, it would result in some excerpts do not have appropriate beginnings, middles, and ends (see Section 7.5.3), the generated outputs can be improved by applying segmentation methods (Rafael et al. 2009; Rodríguez-López and Volk 2013) to data preprocessing.

What musical features can be extracted from music data to represent the quality of various musical dimensions?

Finally, to address research question 3, I investigate correlations between the collected ratings in the six musical dimensions and musical features extracted from the corresponding excerpts. These musical features include statistical complexity, translational complexity, arc score, tonal ambiguity, time intervals and onset jitters. To extend the low-level features (Statistics calculated using only a single factor, for example, pitch classes distribution) that have been commonly used in previous work (Eerola and Toiviainen 2001; Sturm and Ben-Tal 2017; Dong et al. 2018; Yang and Lerch 2020), I utilise advanced features like statistical complexity (Crutchfield 1994), translational complexity (Foubert et al. 2017) and arc score that measures the geometric form of melody. By making the Raincloud plots (Allen et al. 2019) for ratings and features, I have the following key findings:

1. Statistical complexity of human-composed music remains at one level, while statistical complexity of the generated excerpts can be higher or lower than that level. According to fact human-composed music always receives higher ratings, statistical complexity is able to indicate the most appropriate randomness of music in the sequential form.

2. Translational complexity is highly correlated to stylistic success and repetition ratings.

3. Arc score does not have a strong correlation to any music dimension, but demonstrates the expected capability of measuring arc shape.

4. Tonal ambiguity is strongly correlated to stylistic success and melody ratings.

5. IOI is correlated with rhythm ratings for both CSQ and CPI, but KOT/KDT only shows a significant correlation to CSQ/CPI;

6. Onset jitters predict stylistic success and aesthetic pleasure ratings for both CSQ and CPI but in an opposite way due to expressive timing.

As mentioned, feature extraction in most of the previous work has focused on computing statistics on a single musical element, which is often used for comparing characteristics between corpora (Yang and Lerch 2020), and classifying styles or emotions, but does not necessarily describe the quality of music. Here I introduce more advanced features and demonstrate their capability of predicting the quality in various musical dimensions, by utilising the ratings collected in listening study (see Chapter 7). Future work is listed as follows:

1. With the collected ratings of excerpts and these features, research on building ratings prediction models can be conducted. Such a prediction model provides subjective evaluation by learning from human ratings, and saves time and effort of having a listening study. It also provides a greater degree of standardisation and allows direct comparison between generative systems. Furthermore, it is beneficial to investigate the feasibility of embedding such a model as part of the loss function to reinforce the training process and improve output quality.

2. Regarding the presented work, the correlation of statistical complexity to specific musical dimensions can be further investigated. Although

arc score demonstrates its ability to measure melody shape, it remains to be found why there is no significant correlation with the musical dimension.

The entire paper provides a potential framework for improving the automatic evaluation of AMG systems, to overcome the failures stated in Chapter 6. Originality measures are explored in the same Chapter 6; Chapter 7 identifies what human experts use when making judgments of musical quality; Chapter 8 leverages the findings of Chapter 7 to propose new objectively measurable features for automatic evaluation in future work. This proposed new framework for automatic evaluation is an additional contribution to the field of AMG, I will continue and invoke more research work in this direction to improve the deficiencies mentioned earlier.

Appendices

A

Instance of representations

This appendix shows the whole set of MusicXML code introduced in Section 2.2.2, and this code is corresponding to the example piece shown in Figure 2.3.

A.1 MusicXML

```
<score-partwise version="3.1">
  <part-list>
    <score-part id="P1">
      <part-name>Piano</part-name>
    </part-list>
  <part id="P1">
    <measure number="0">
      <attributes>
        <divisions>4</divisions>
        <key>
          <fifths>-1</fifths>
          </key>
        <time>
          <beats>2</beats>
          <beat-type>4</beat-type>
          </time>
        <staves>2</staves>
        <clef number="1">
          <sign>G</sign>
          <line>2</line>
          </clef>
        <clef number="2">
```

```
23              <sign>F</sign>
24              <line>4</line>
25              </clef>
26            </attributes>
27          <note>
28            <rest/>
29            <duration>2</duration>
30            <voice>1</voice>
31            <type>eighth</type>
32            <staff>1</staff>
33            </note>
34          <backup>
35            <duration>2</duration>
36            </backup>
37          <note>
38            <pitch>
39              <step>B</step>
40              <alter>-1</alter>
41              <octave>1</octave>
42              </pitch>
43            <duration>1</duration>
44            <voice>5</voice>
45            <type>16th</type>
46            <stem>up</stem>
47            <staff>2</staff>
48            <beam number="1">begin</beam>
49            <beam number="2">begin</beam>
50            </note>
51          <note>
52            <chord/>
53            <pitch>
54              <step>B</step>
55              <alter>-1</alter>
56              <octave>2</octave>
57              </pitch>
58            <duration>1</duration>
59            <voice>5</voice>
60            <type>16th</type>
61            <stem>up</stem>
62            <staff>2</staff>
63            </note>
64          <note>
65            <pitch>
```

```
66              <step>B</step>
67              <alter>-1</alter>
68              <octave>1</octave>
69              </pitch>
70            <duration>1</duration>
71            <voice>5</voice>
72            <type>16th</type>
73            <stem>up</stem>
74            <staff>2</staff>
75            <beam number="1">end</beam>
76            <beam number="2">end</beam>
77            </note>
78          <note>
79            <chord/>
80            <pitch>
81              <step>B</step>
82              <alter>-1</alter>
83              <octave>2</octave>
84              </pitch>
85            <duration>1</duration>
86            <voice>5</voice>
87            <type>16th</type>
88            <stem>up</stem>
89            <staff>2</staff>
90            </note>
91          </measure>
92        <measure number="1">
93          <note>
94            <pitch>
95              <step>F</step>
96              <octave>3</octave>
97              </pitch>
98            <duration>8</duration>
99            <voice>1</voice>
100           <type>half</type>
101           <stem>up</stem>
102           <staff>1</staff>
103           </note>
104         <note>
105           <chord/>
106           <pitch>
107             <step>A</step>
108             <octave>3</octave>
```

```
109                 </pitch>
110               <duration>8</duration>
111               <voice>1</voice>
112               <type>half</type>
113               <stem>up</stem>
114               <staff>1</staff>
115               </note>
116             <note>
117               <chord/>
118               <pitch>
119                 <step>B</step>
120                 <alter>-1</alter>
121                 <octave>3</octave>
122                 </pitch>
123               <duration>8</duration>
124               <voice>1</voice>
125               <type>half</type>
126               <stem>up</stem>
127               <staff>1</staff>
128               </note>
129             <note>
130               <chord/>
131               <pitch>
132                 <step>D</step>
133                 <octave>4</octave>
134                 </pitch>
135               <duration>8</duration>
136               <voice>1</voice>
137               <type>half</type>
138               <stem>up</stem>
139               <staff>1</staff>
140               </note>
141             <backup>
142               <duration>8</duration>
143               </backup>
144             <note>
145               <pitch>
146                 <step>B</step>
147                 <alter>-1</alter>
148                 <octave>1</octave>
149                 </pitch>
150               <duration>4</duration>
151               <voice>5</voice>
```

```
152             <type>quarter</type>
153             <stem>up</stem>
154             <staff>2</staff>
155             </note>
156           <note>
157             <chord/>
158             <pitch>
159               <step>B</step>
160               <alter>-1</alter>
161               <octave>2</octave>
162               </pitch>
163             <duration>4</duration>
164             <voice>5</voice>
165             <type>quarter</type>
166             <stem>up</stem>
167             <staff>2</staff>
168             </note>
169           <note>
170             <pitch>
171               <step>B</step>
172               <alter>-1</alter>
173               <octave>2</octave>
174               </pitch>
175             <duration>3</duration>
176             <voice>5</voice>
177             <type>eighth</type>
178             <dot/>
179             <stem>up</stem>
180             <staff>2</staff>
181             <beam number="1">begin</beam>
182             </note>
183           <note>
184             <pitch>
185               <step>B</step>
186               <alter>-1</alter>
187               <octave>1</octave>
188               </pitch>
189             <duration>1</duration>
190             <voice>5</voice>
191             <type>16th</type>
192             <stem>up</stem>
193             <staff>2</staff>
194             <beam number="1">end</beam>
```

```
195             <beam number="2">backward hook</beam>
196             </note>
197           </measure>
198         </part>
199       </score-partwise>
```

B

Statistical analysis

This appendix contains Raincloud plots (Allen et al. 2019) for all pairs of features and musical dimensions discussed in Section 8.2.

B.1 Classical string quartets

B.1.1 Translational Complexity

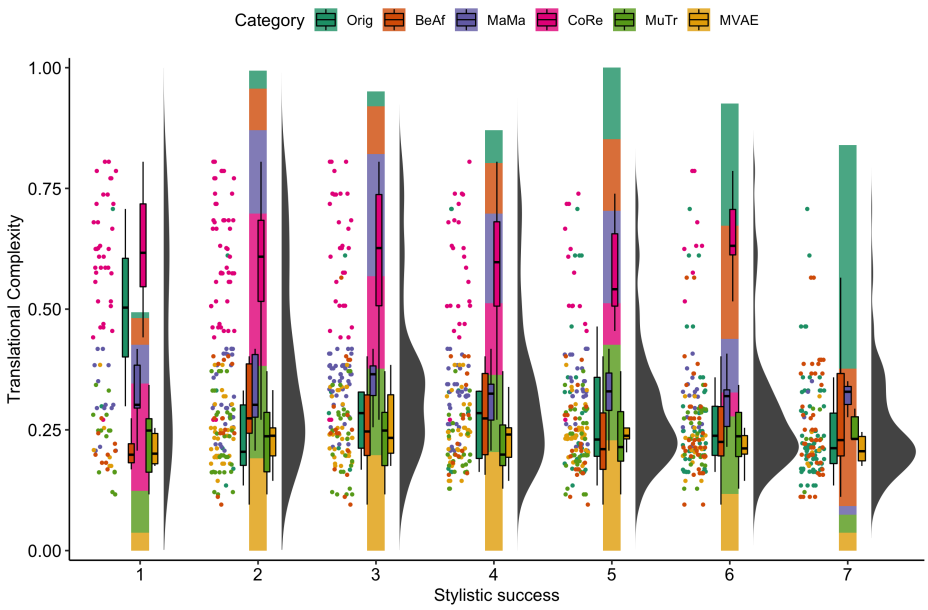


Figure B.1: Translational Complexity against Stylistic success ratings.

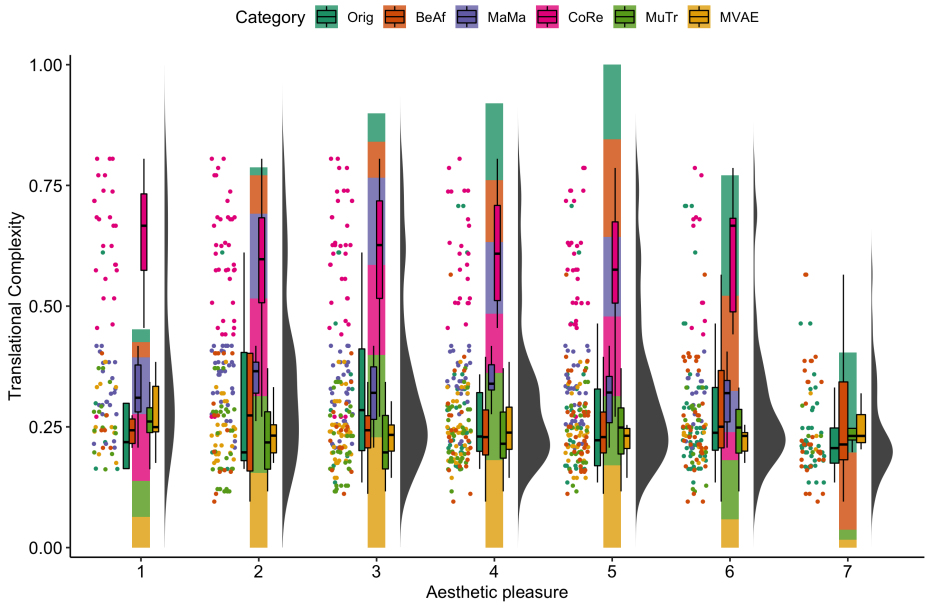


Figure B.2: Translational Complexity against Aesthetic pleasure ratings.

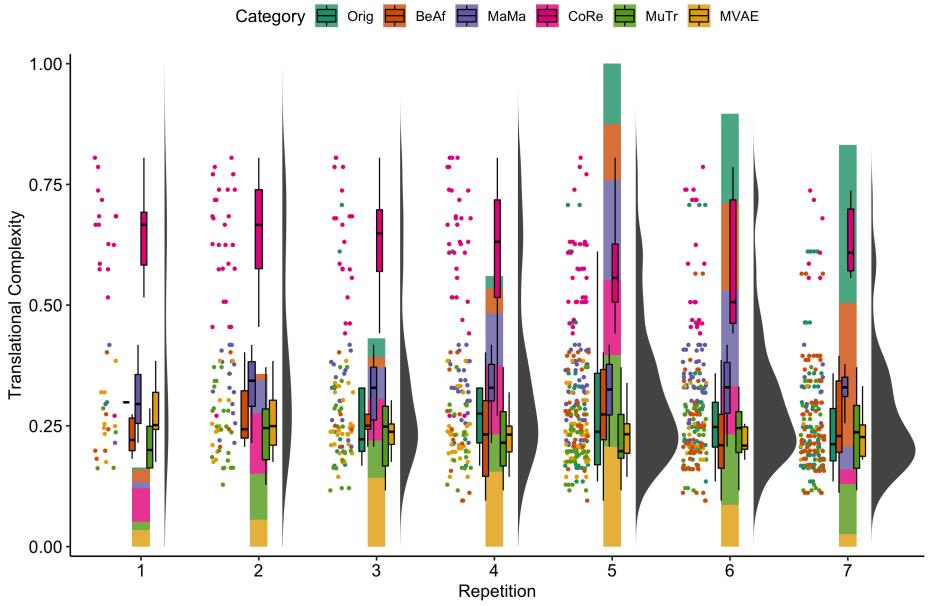


Figure B.3: Translational Complexity against Repetition ratings.

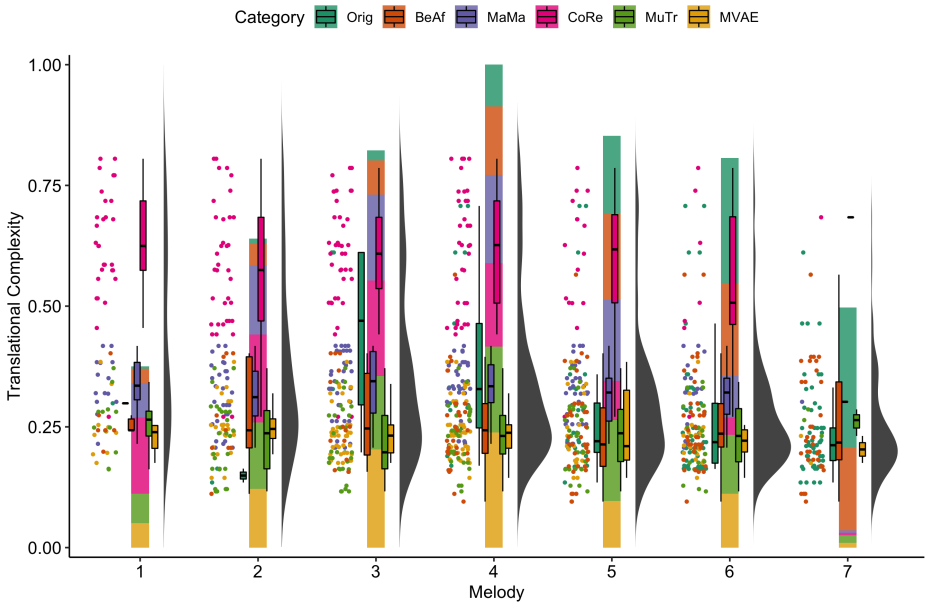


Figure B.4: Translational Complexity against Melody ratings.

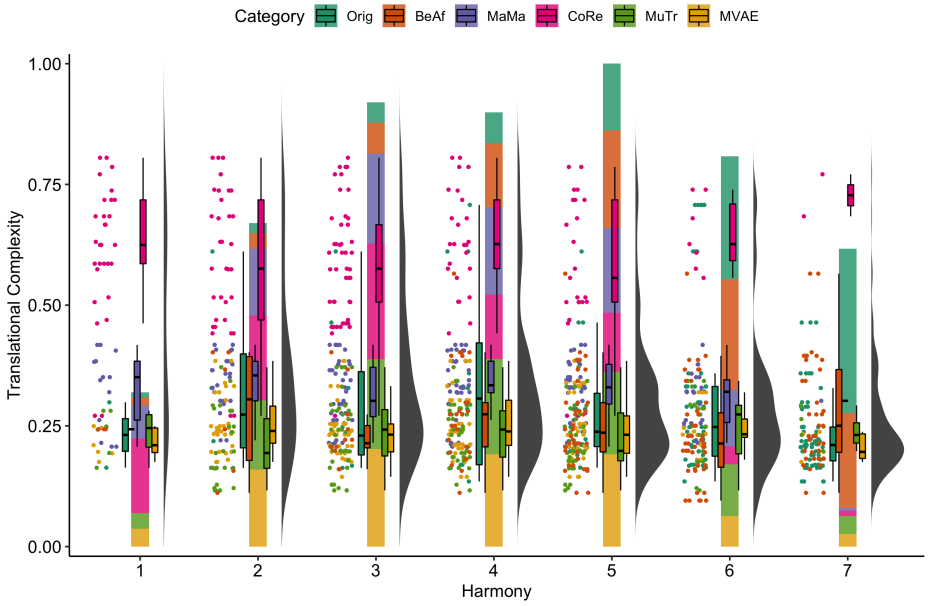


Figure B.5: Translational Complexity against Harmony ratings.

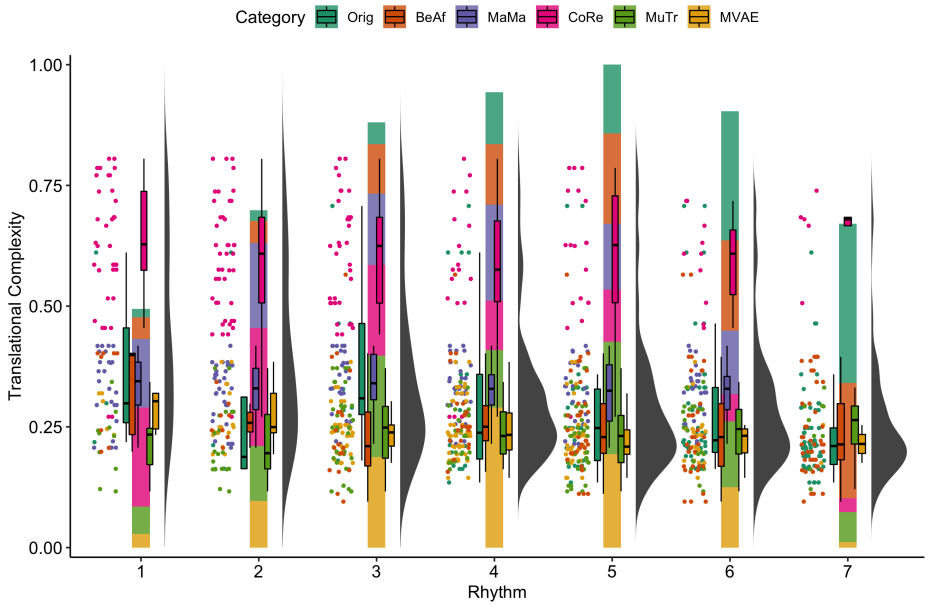


Figure B.6: Translational Complexity against Rhythm ratings.

B.1.2 Arc score

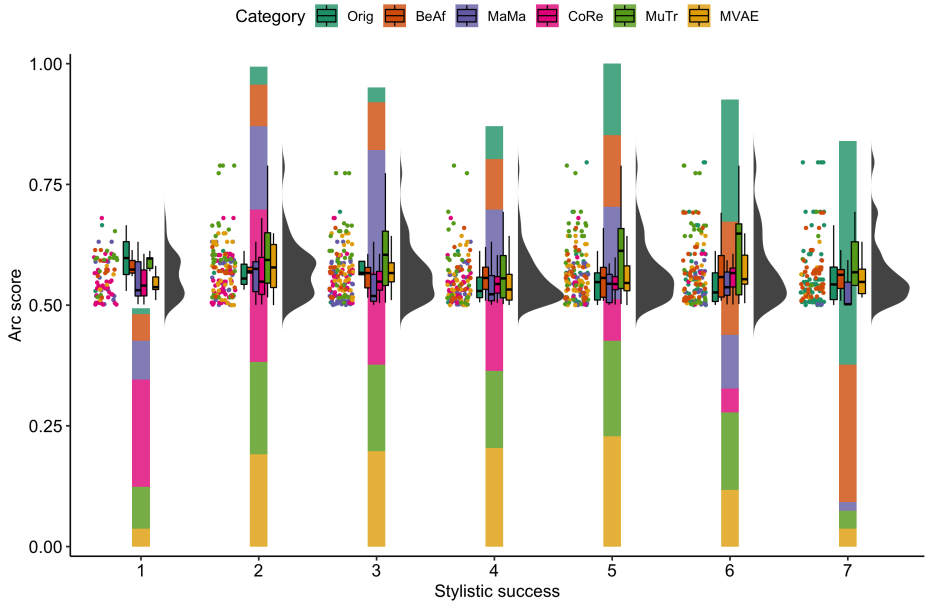


Figure B.7: Arc score against Stylistic success ratings.

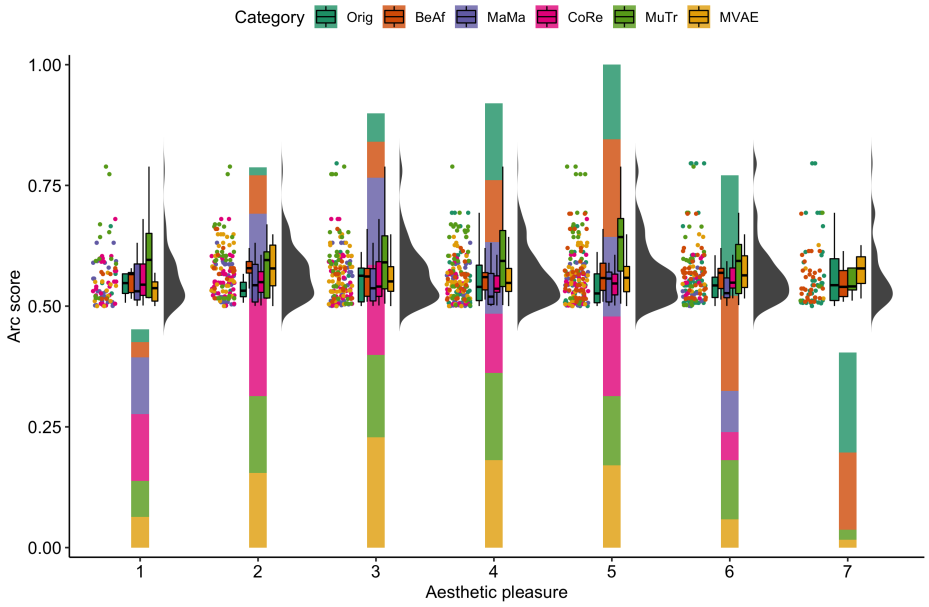


Figure B.8: Arc score against Aesthetic pleasure ratings.

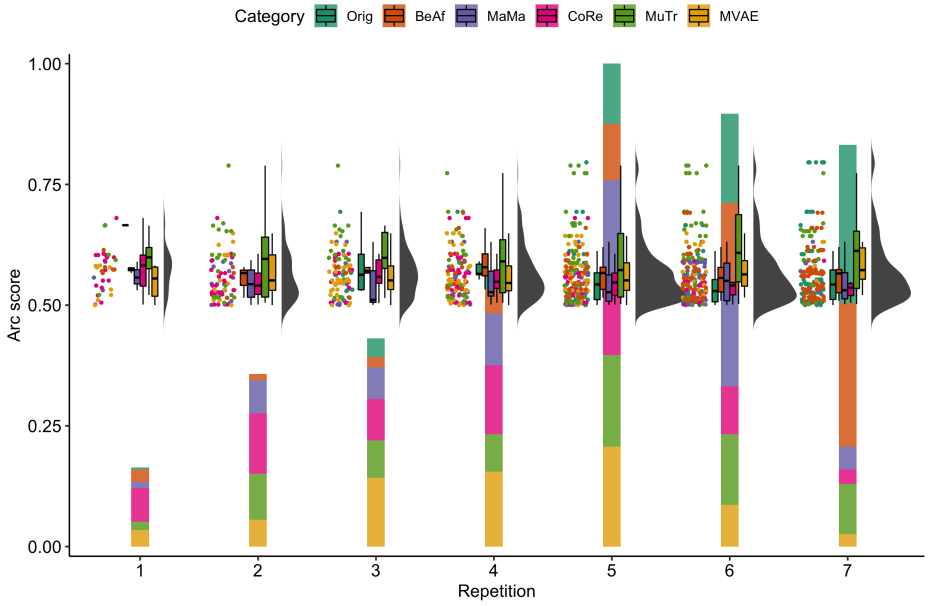


Figure B.9: Arc score against Repetition ratings.

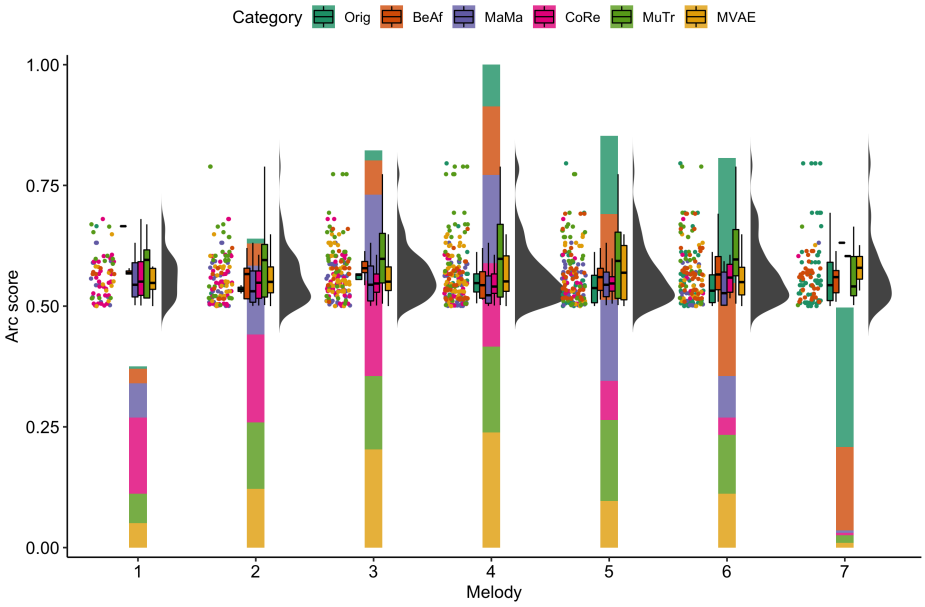


Figure B.10: Arc score against Melody ratings.

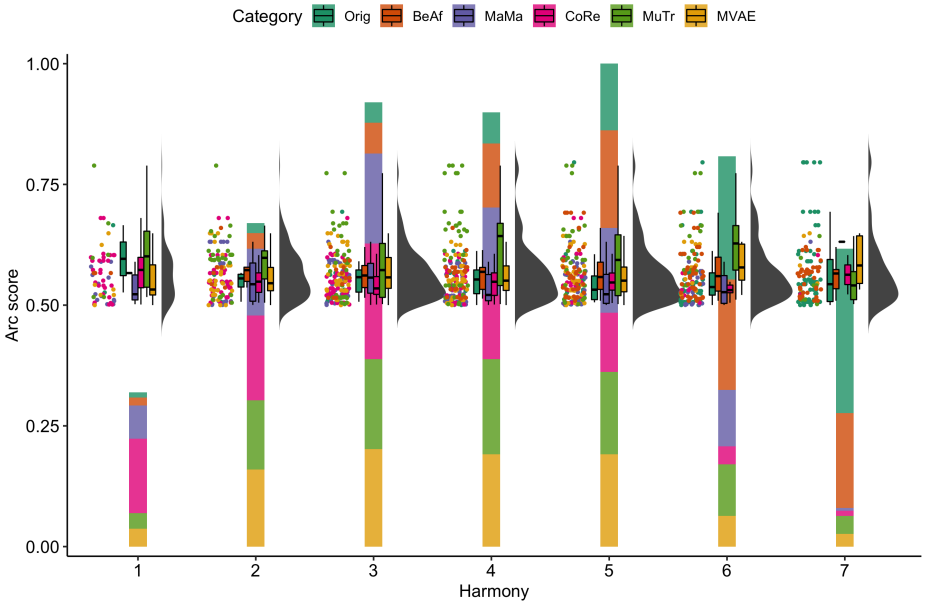


Figure B.11: Arc score against Harmony ratings.

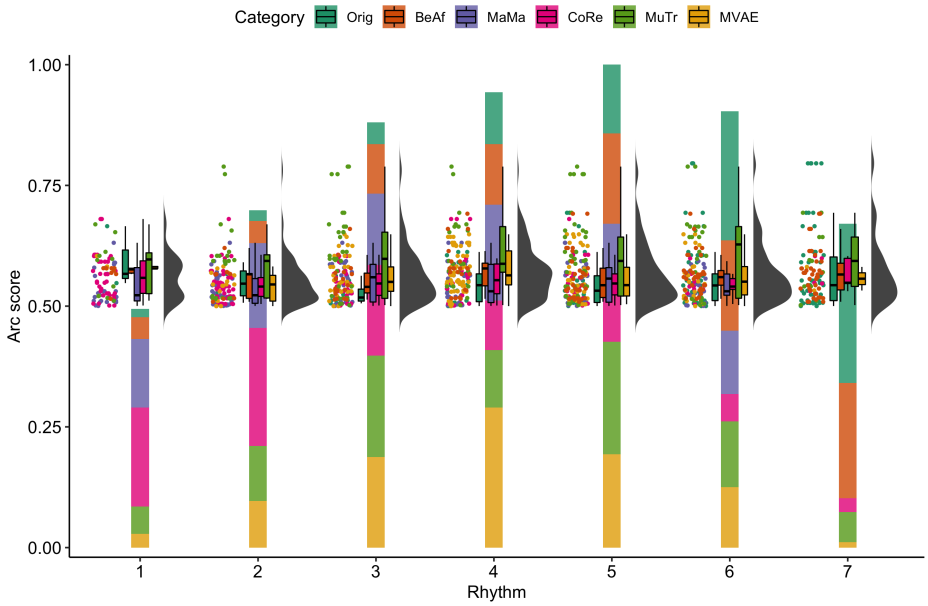


Figure B.12: Arc score against Rhythm ratings.

B.1.3 Tonal ambiguity

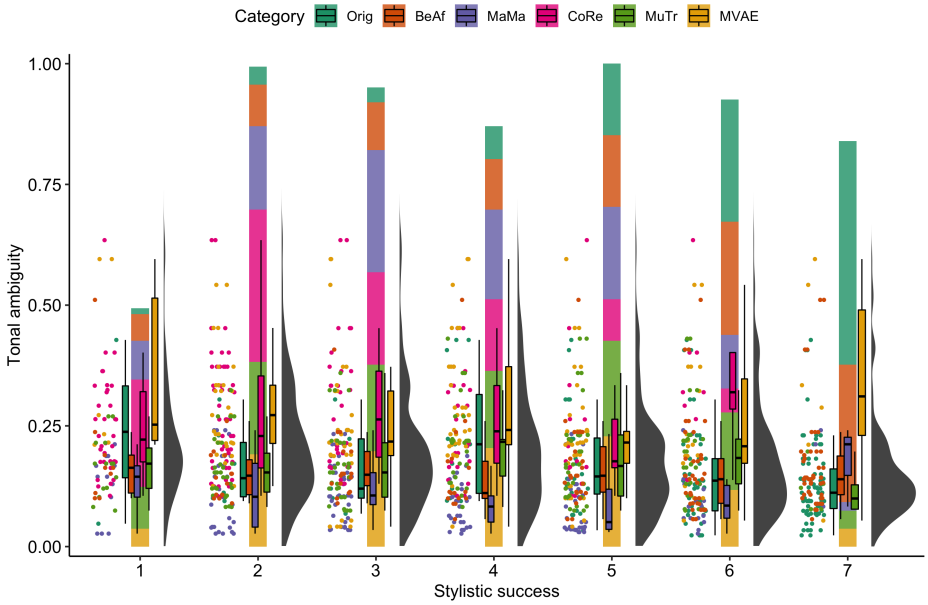


Figure B.13: Tonal ambiguity against Stylistic success ratings.

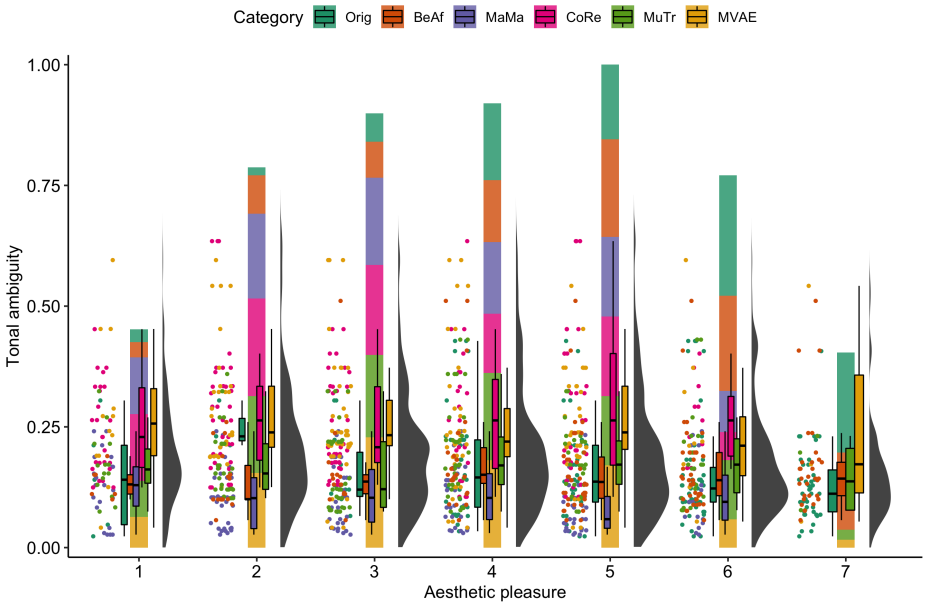


Figure B.14: Tonal ambiguity against Aesthetic pleasure ratings.

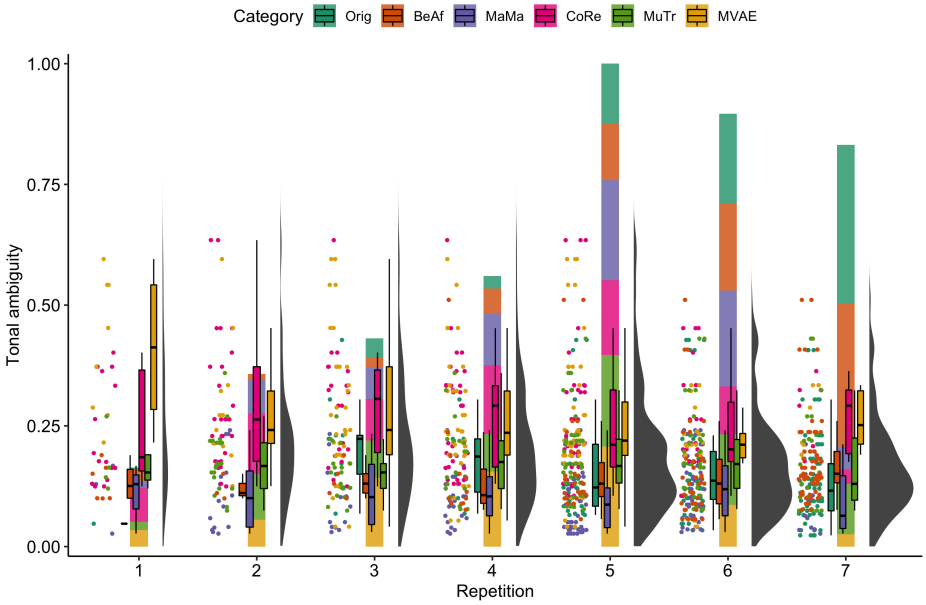


Figure B.15: Tonal ambiguity against Repetition ratings.

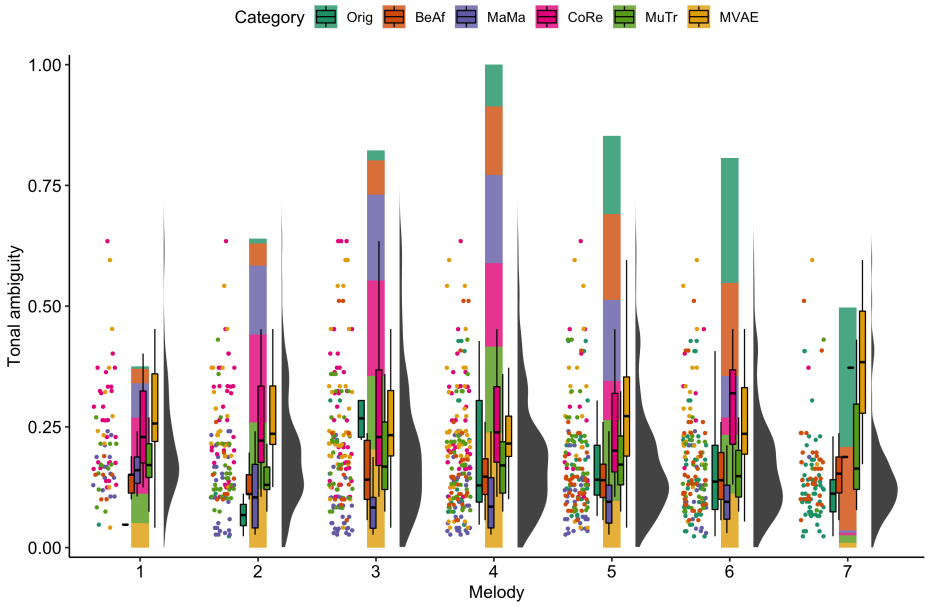


Figure B.16: Tonal ambiguity against Melody ratings.

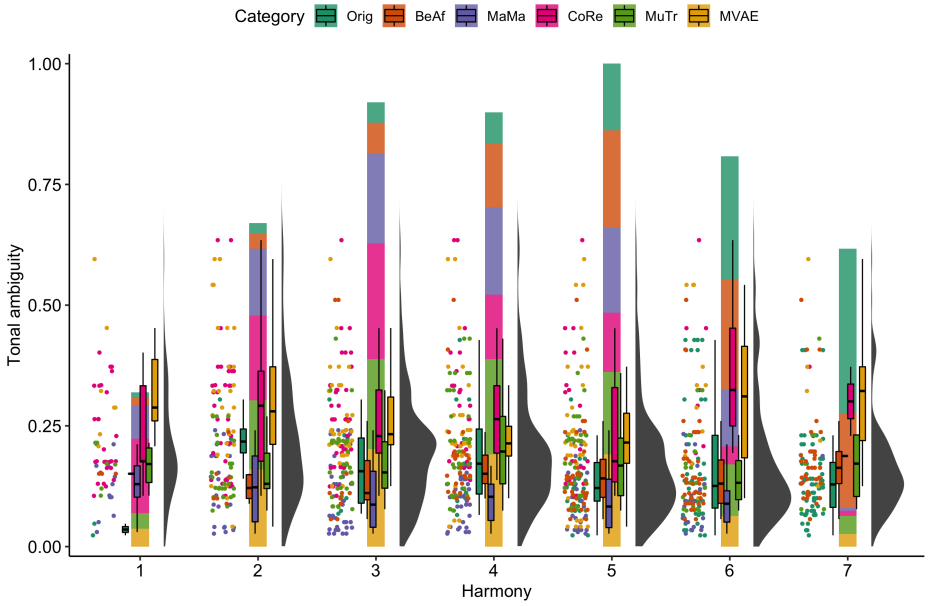


Figure B.17: Tonal ambiguity against Harmony ratings.

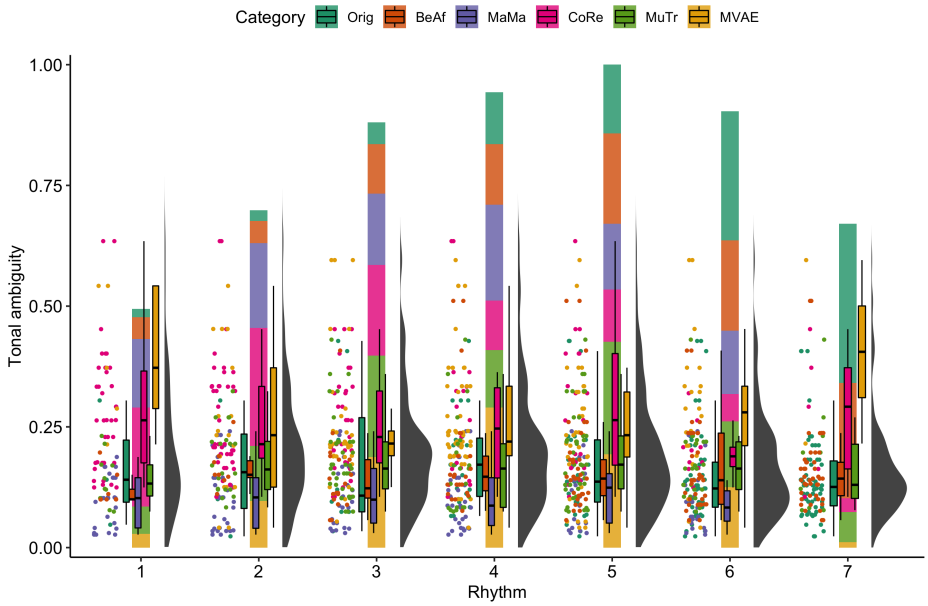


Figure B.18: Tonal ambiguity against Rhythm ratings.

B.1.4 IOI mean

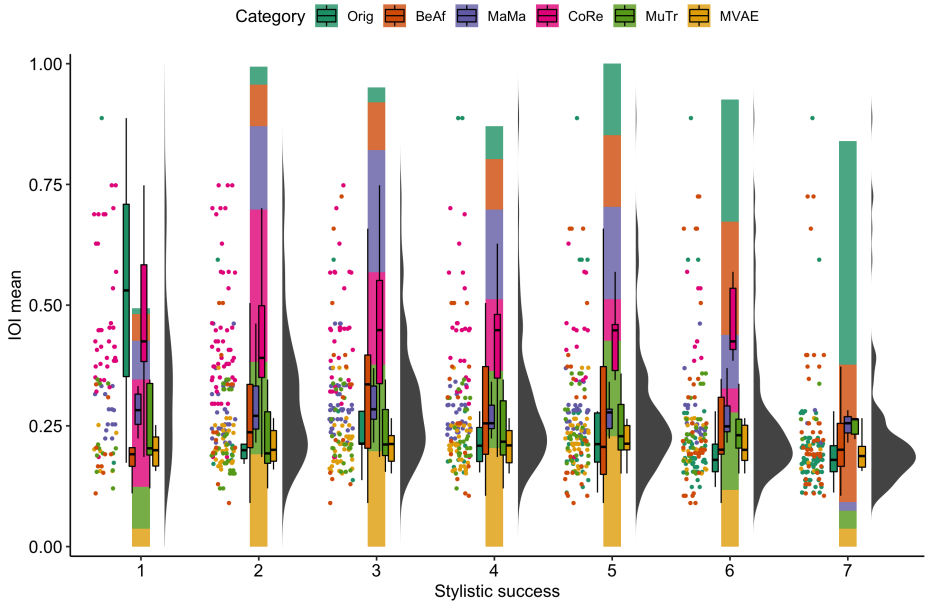


Figure B.19: IOI mean against Stylistic success ratings.

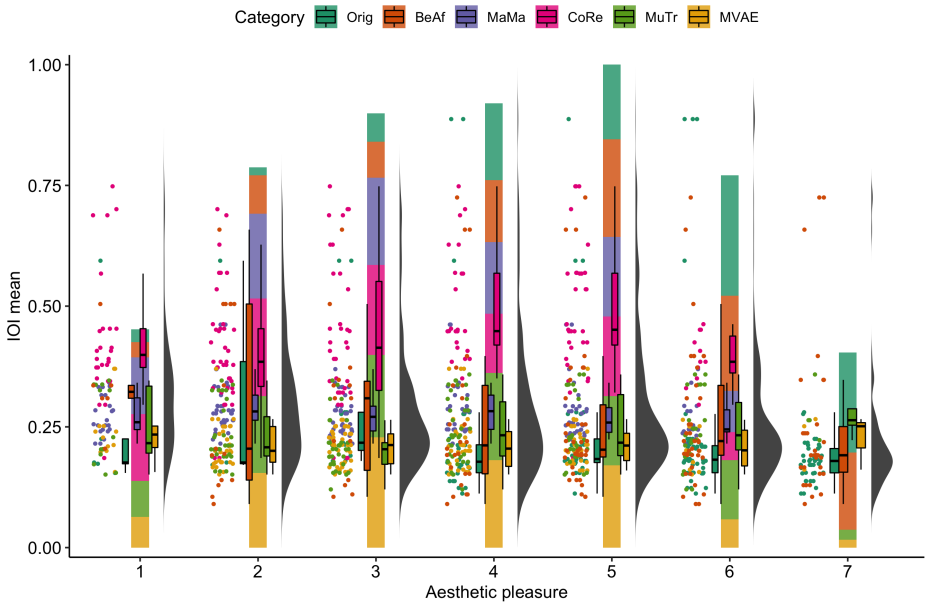


Figure B.20: IOI mean against Aesthetic pleasure ratings.

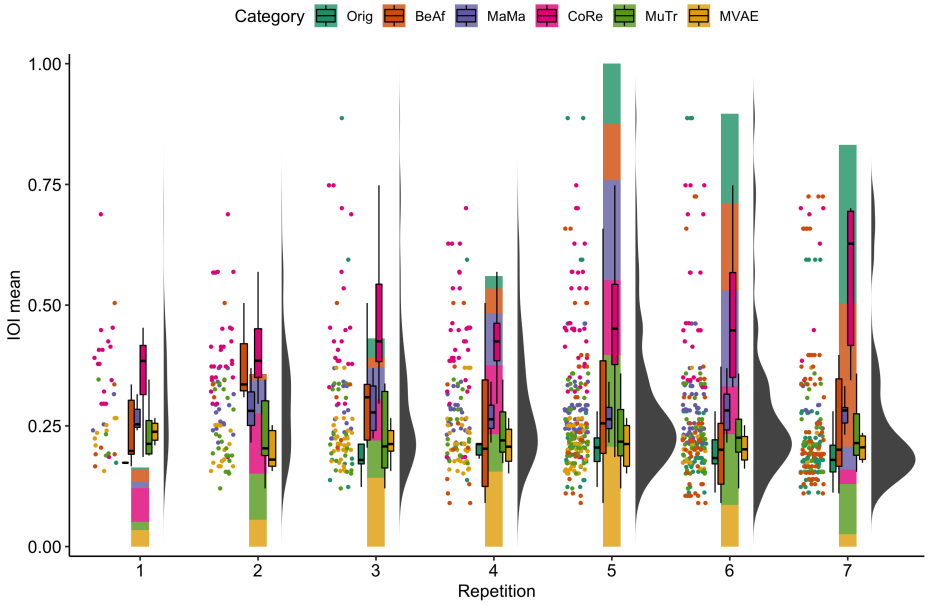


Figure B.21: IOI mean against Repetition ratings.

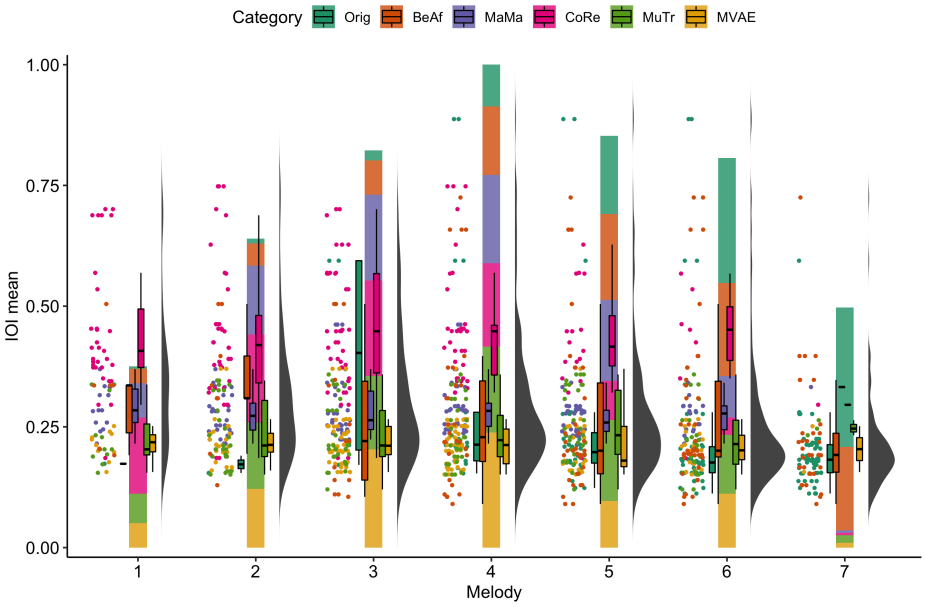


Figure B.22: IOI mean against Melody ratings.

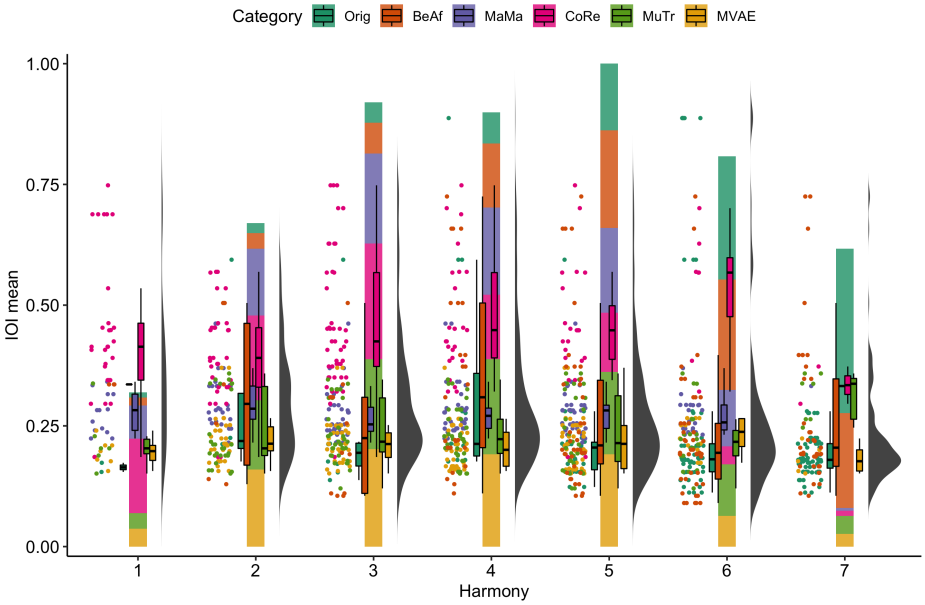


Figure B.23: IOI mean against Harmony ratings.

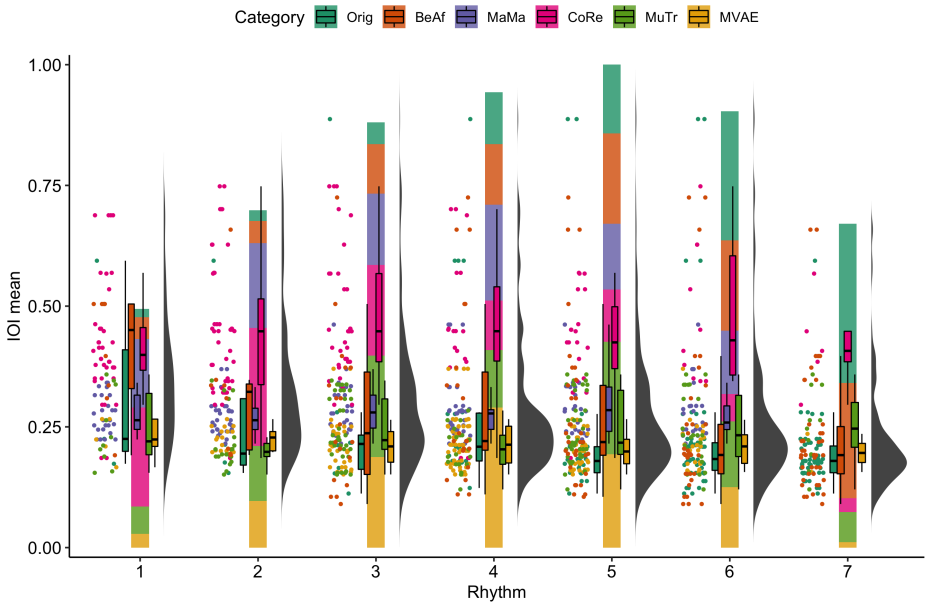


Figure B.24: IOI mean against Rhythm ratings.

B.1.5 IOI variance

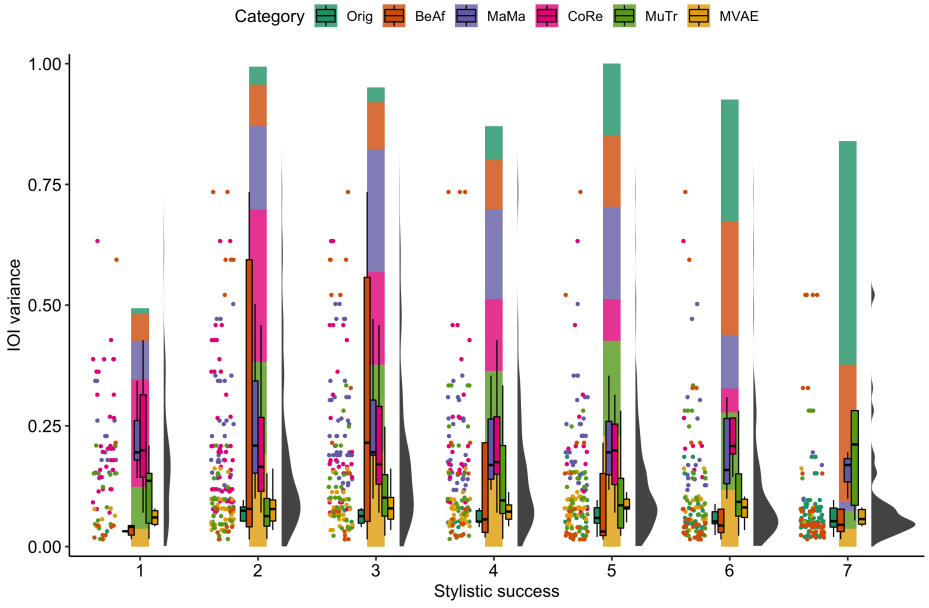


Figure B.25: IOI variance against Stylistic success ratings.

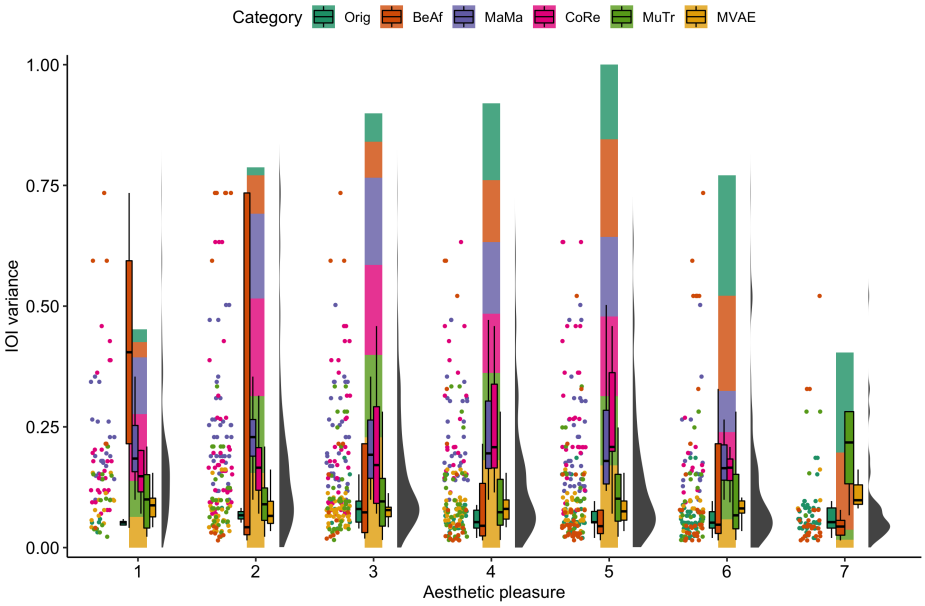


Figure B.26: IOI variance against Aesthetic pleasure ratings.

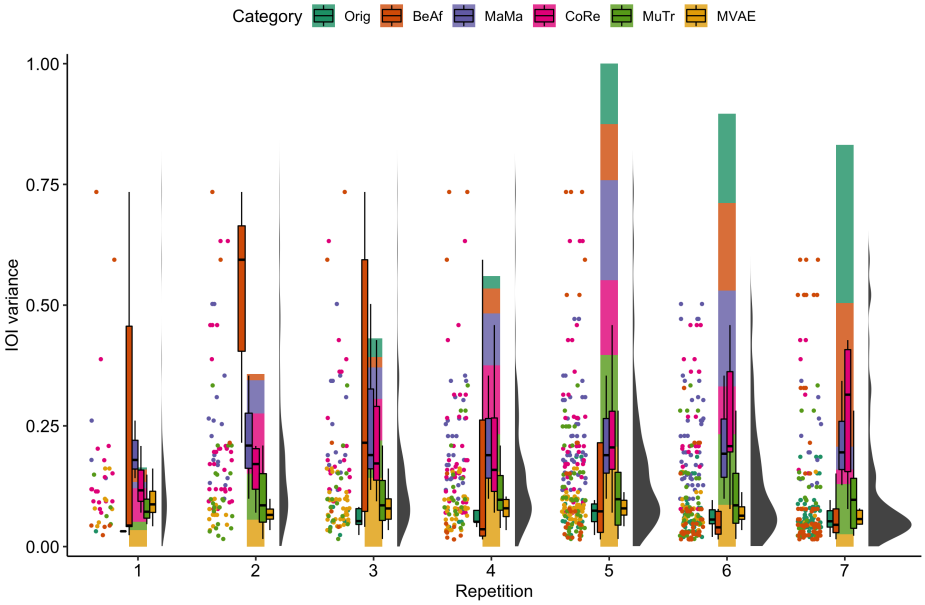


Figure B.27: IOI variance against Repetition ratings.

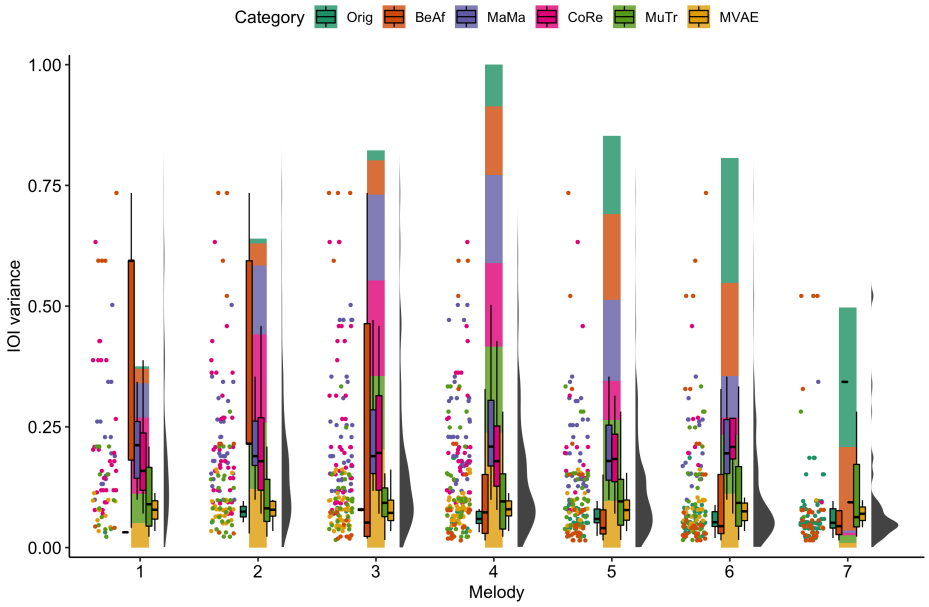


Figure B.28: IOI variance against Melody ratings.

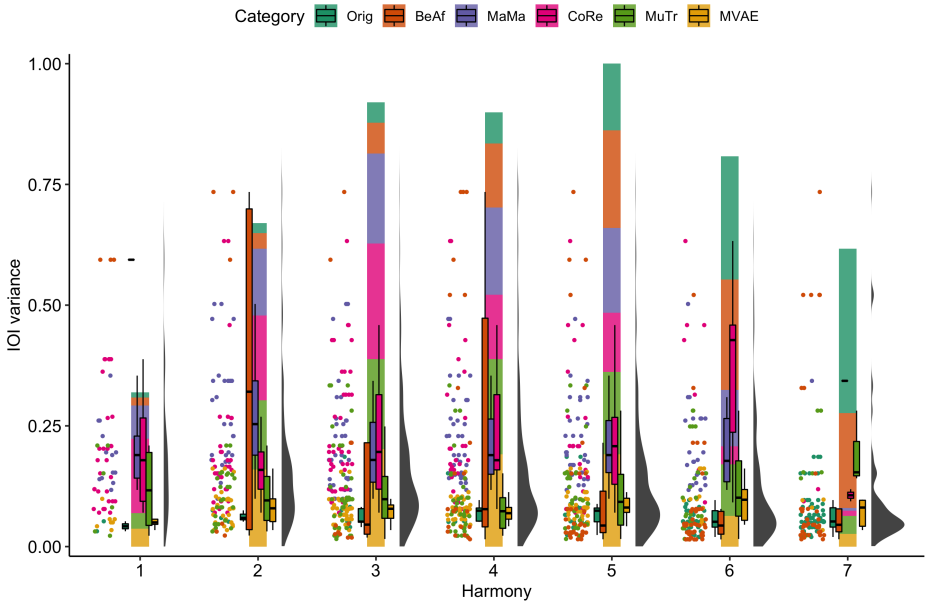


Figure B.29: IOI variance against Harmony ratings.

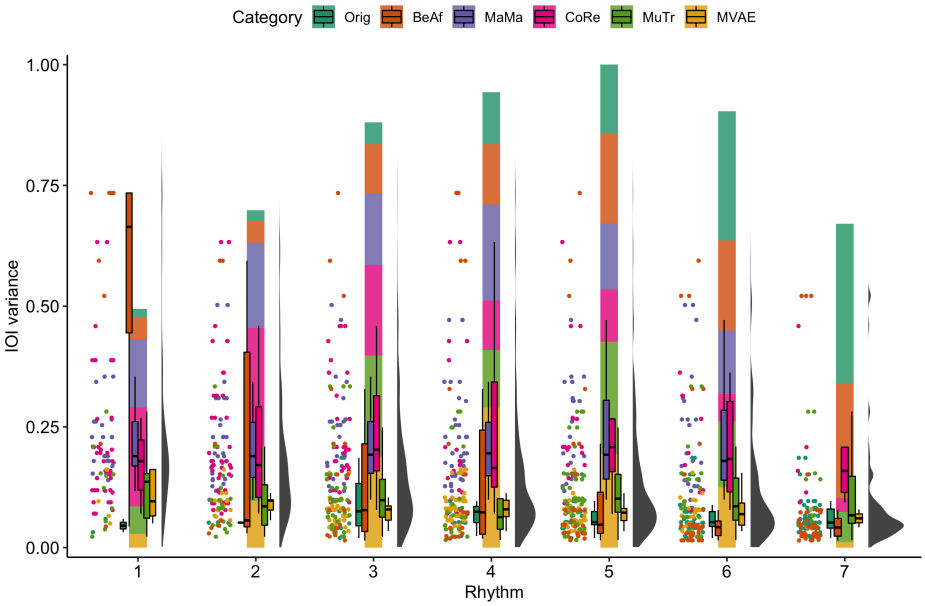


Figure B.30: IOI variance against Rhythm ratings.

B.1.6 IOI2 mean

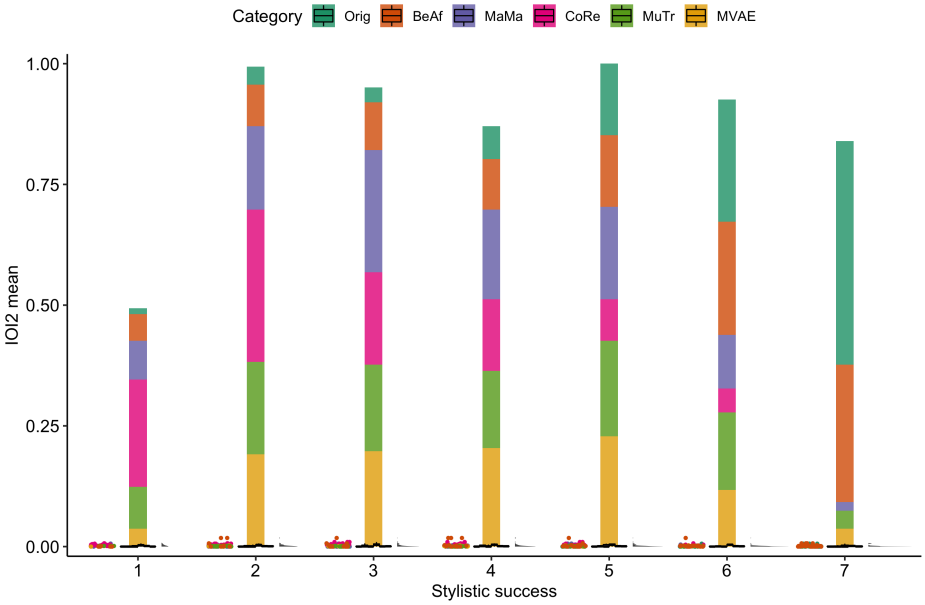


Figure B.31: IOI2 mean against Stylistic success ratings.

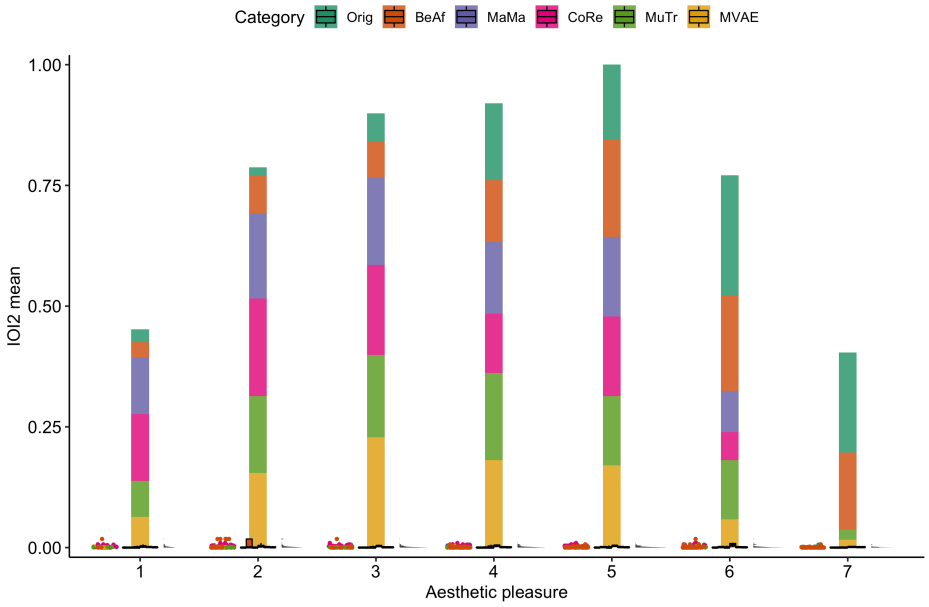


Figure B.32: IOI2 mean against Aesthetic pleasure ratings.

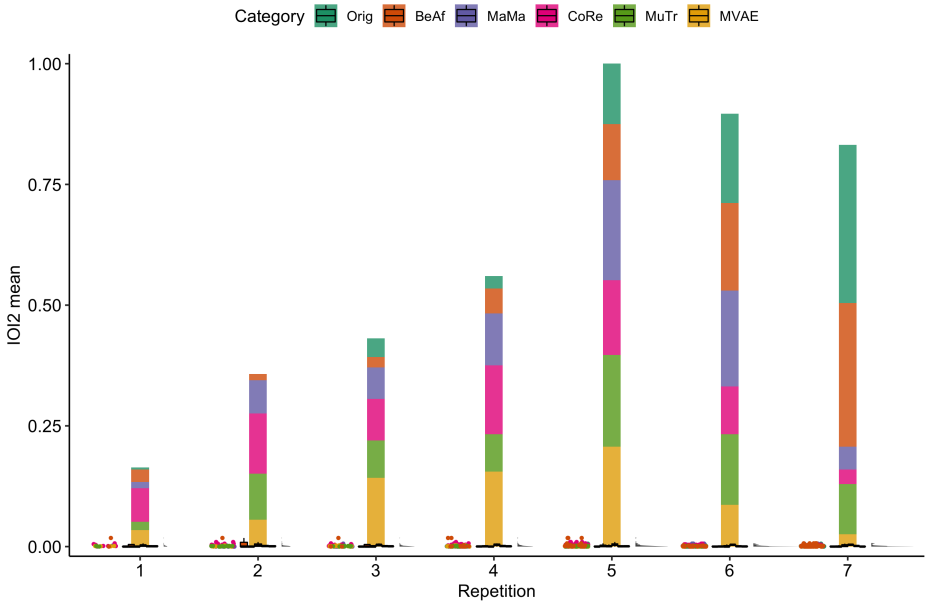


Figure B.33: IOI2 mean against Repetition ratings.

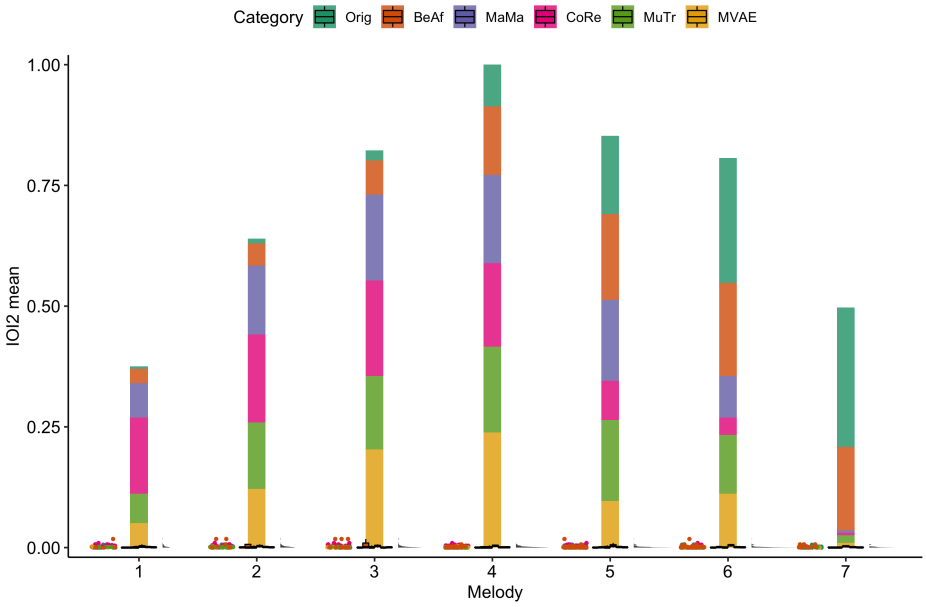


Figure B.34: IOI2 mean against Melody ratings.

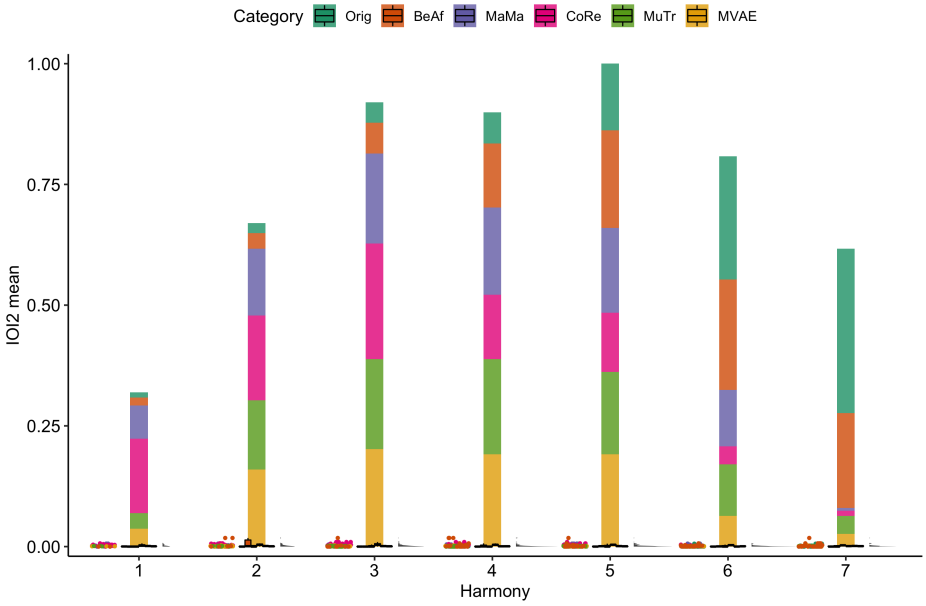


Figure B.35: IOI2 mean against Harmony ratings.

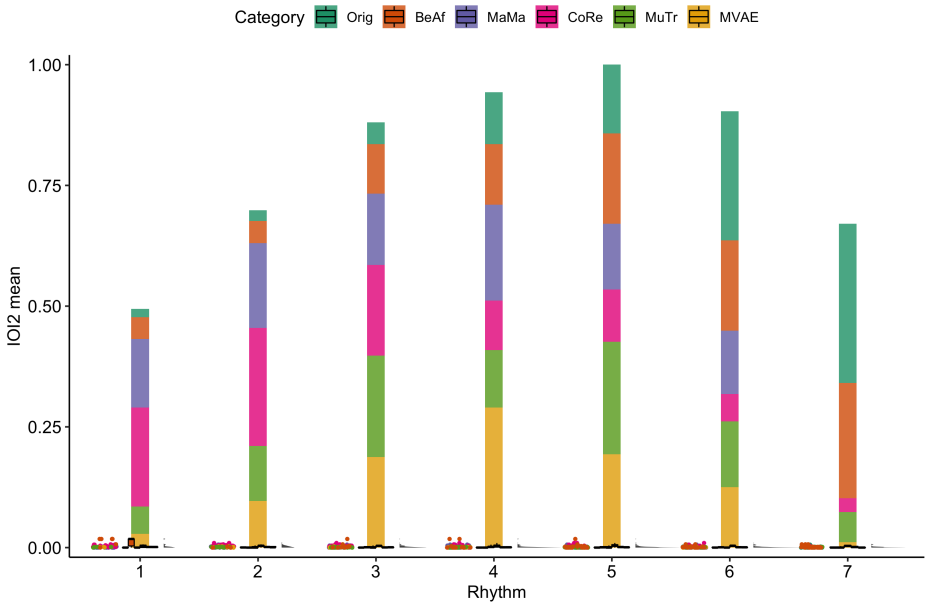


Figure B.36: IOI2 mean against Rhythm ratings.

B.1.7 IOI2 variance

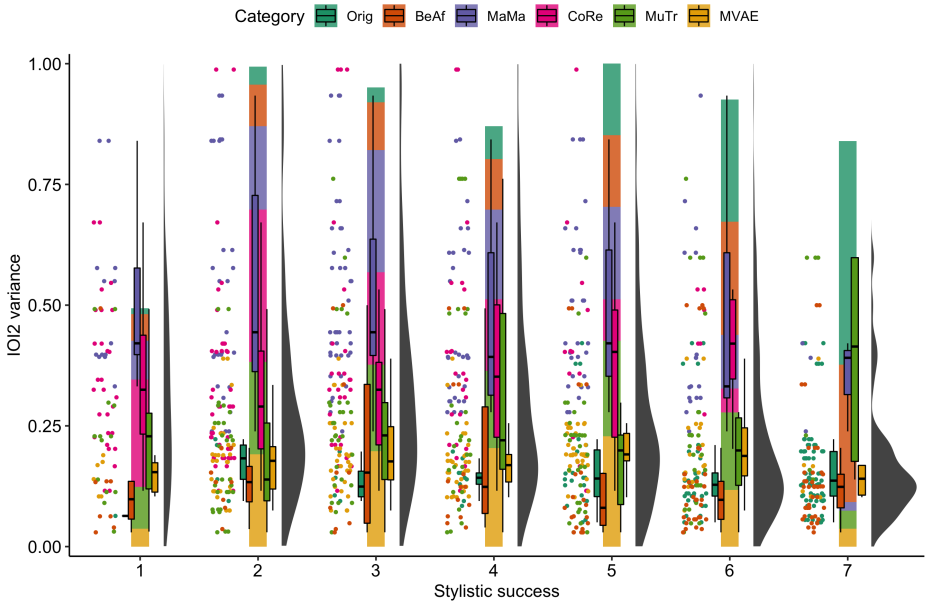


Figure B.37: IOI2 variance against Stylistic success ratings.

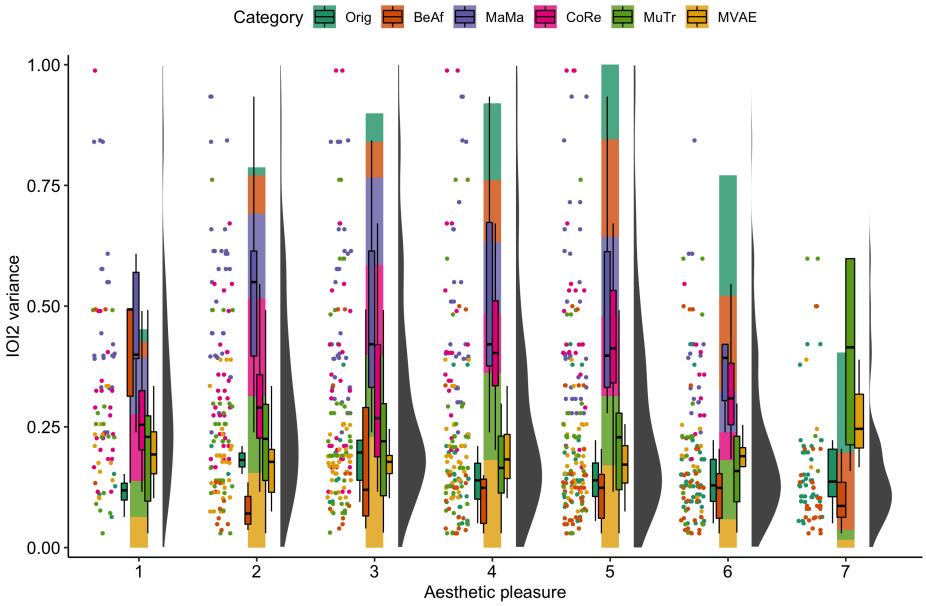


Figure B.38: IOI2 variance against Aesthetic pleasure ratings.

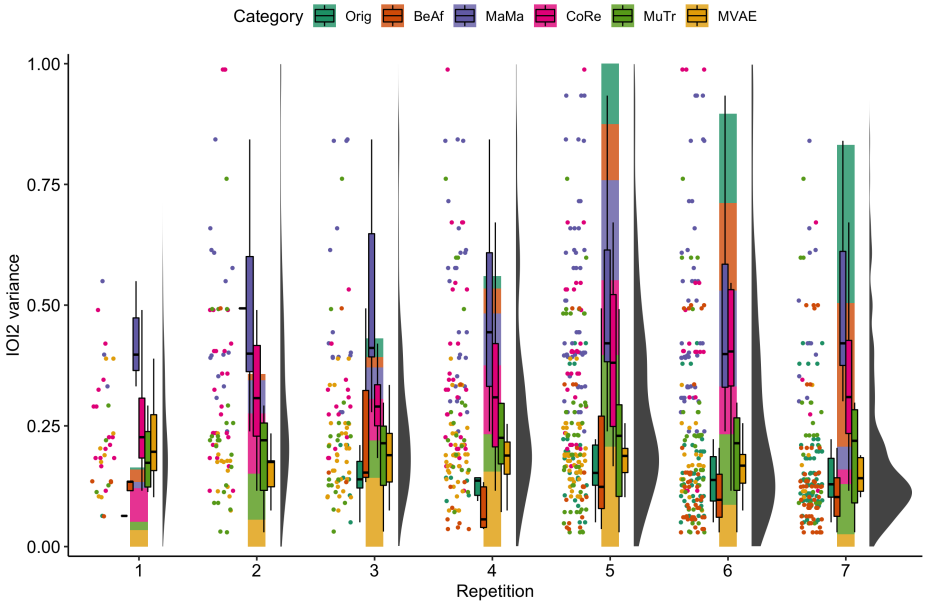


Figure B.39: IOI2 variance against Repetition ratings.

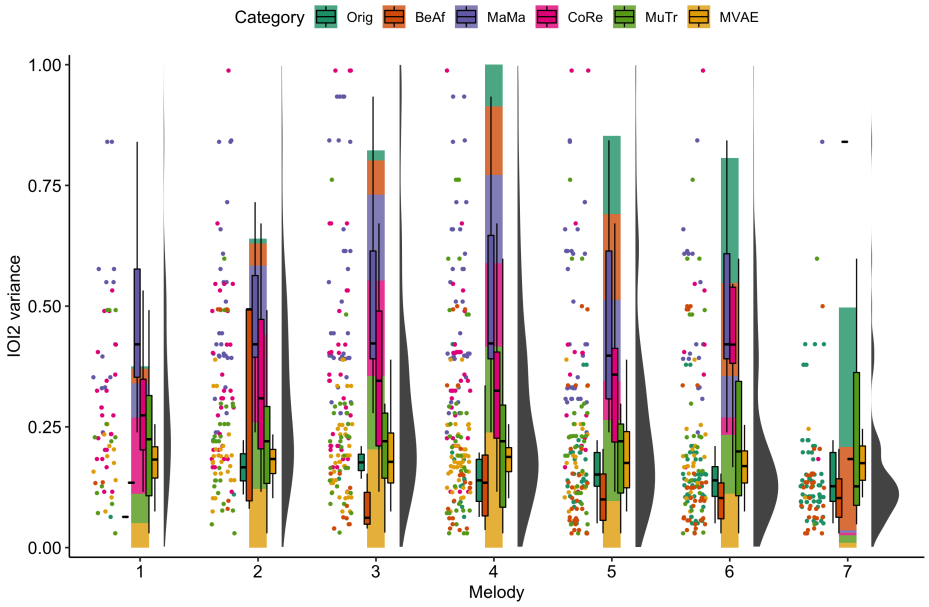


Figure B.40: IOI2 variance against Melody ratings.

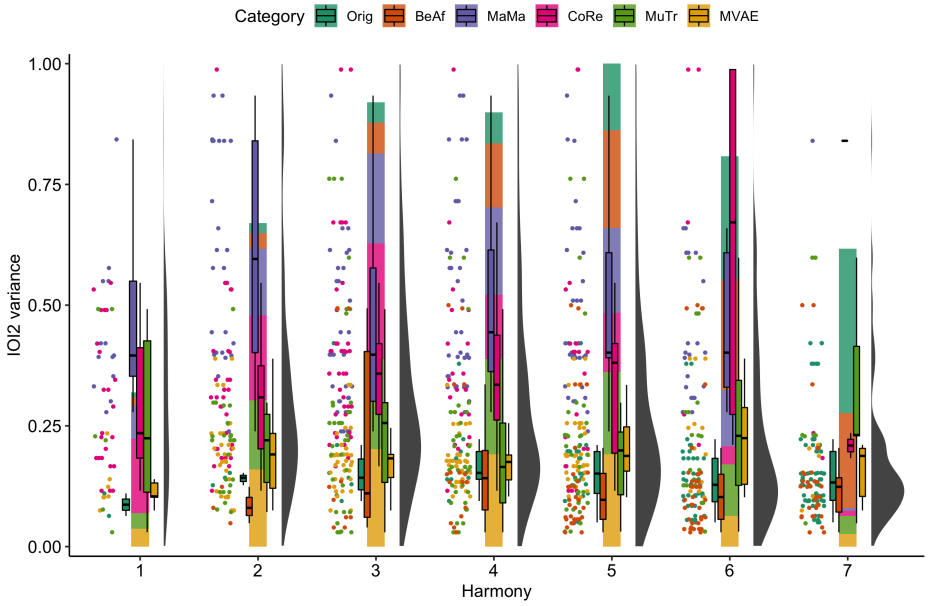


Figure B.41: IOI2 variance against Harmony ratings.

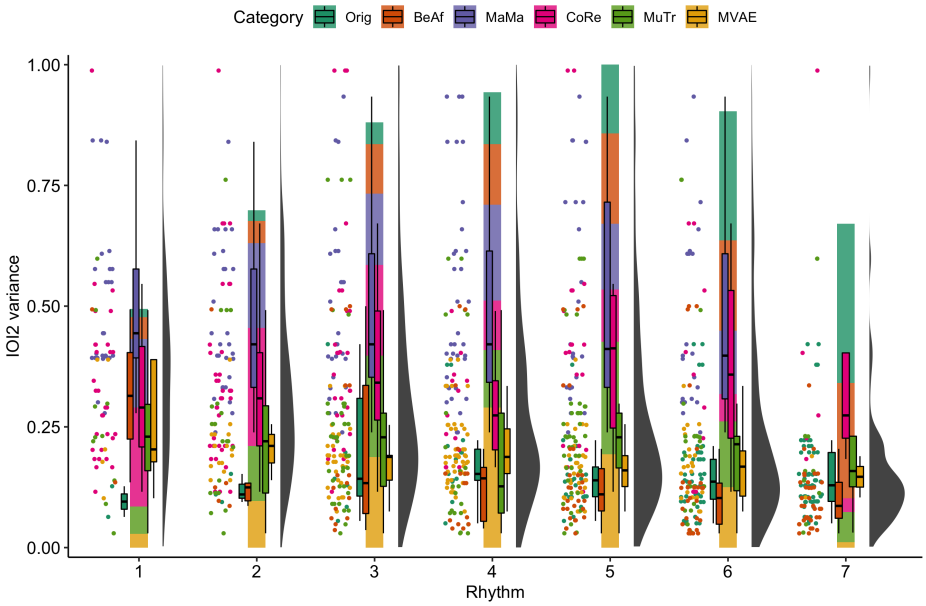


Figure B.42: IOI2 variance against Rhythm ratings.

B.1.8 KOT mean

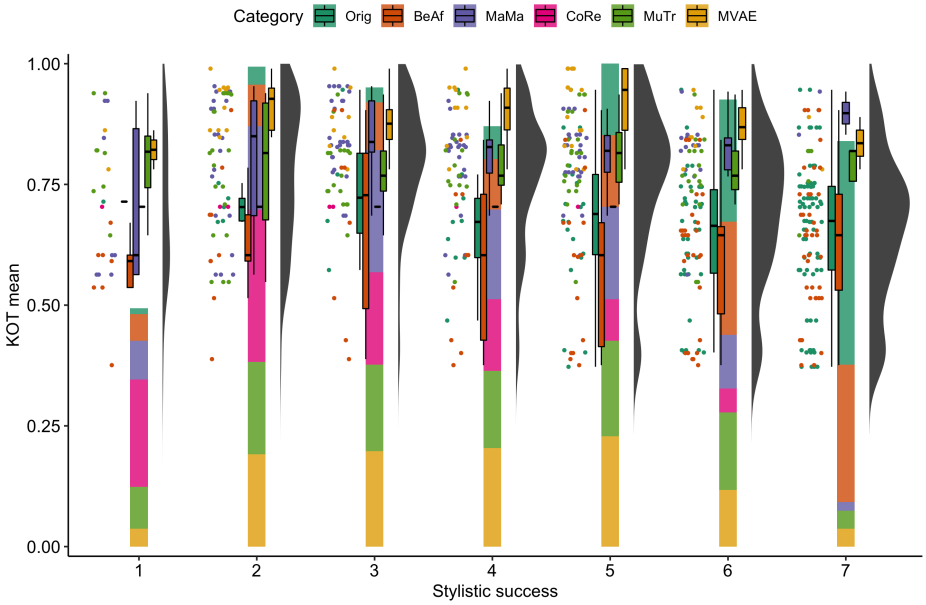


Figure B.43: KOT mean against Stylistic success ratings.

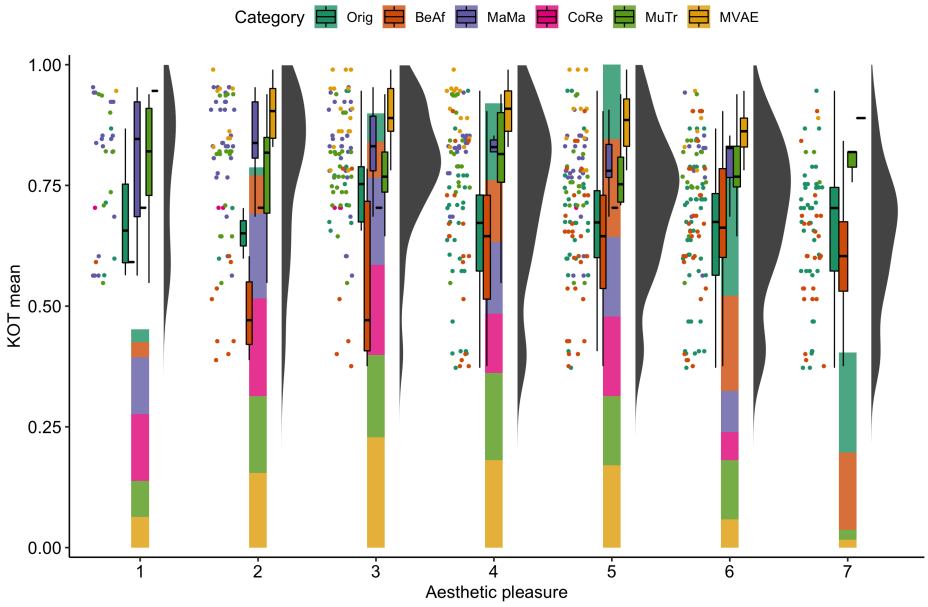


Figure B.44: KOT mean against Aesthetic pleasure ratings.

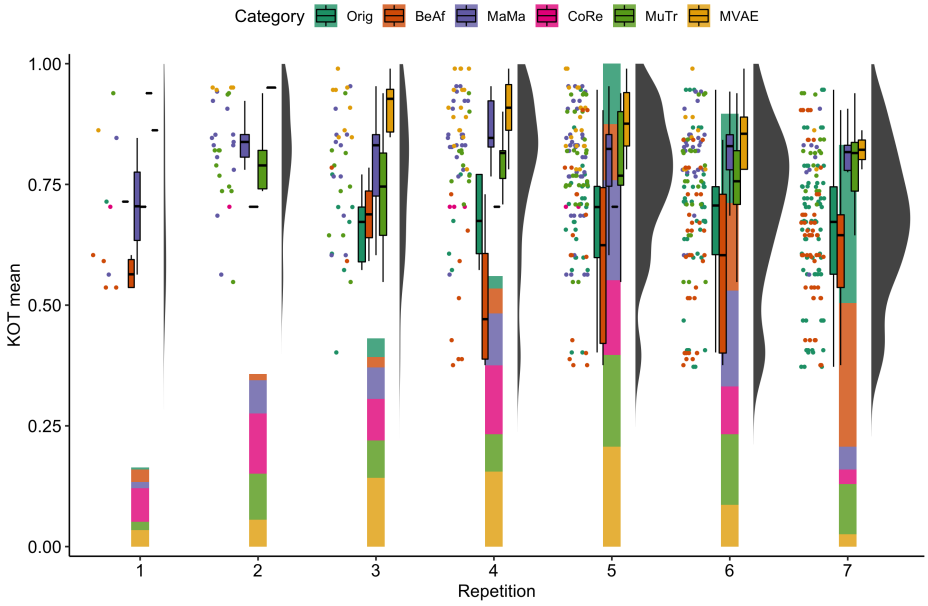


Figure B.45: KOT mean against Repetition ratings.

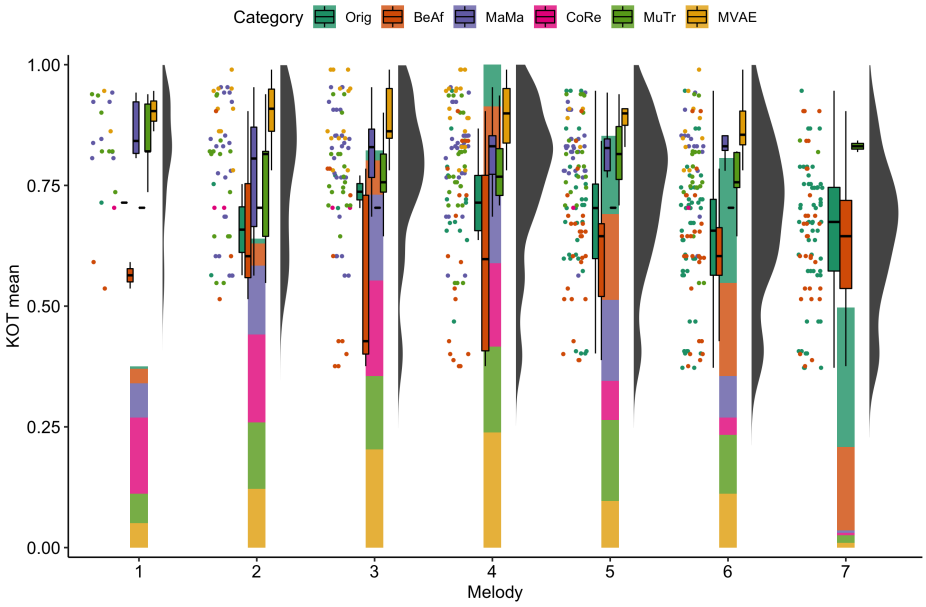


Figure B.46: KOT mean against Melody ratings.

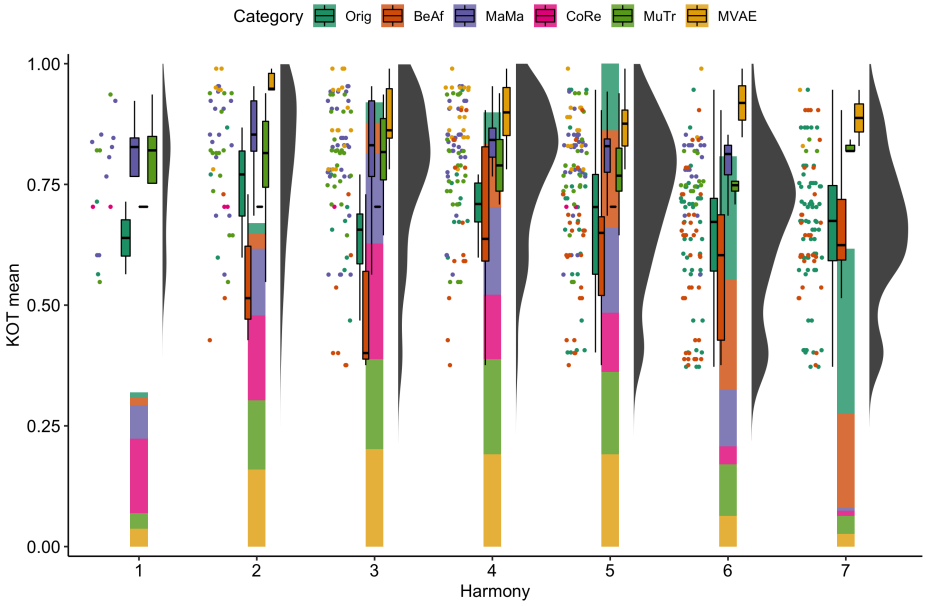


Figure B.47: KOT mean against Harmony ratings.

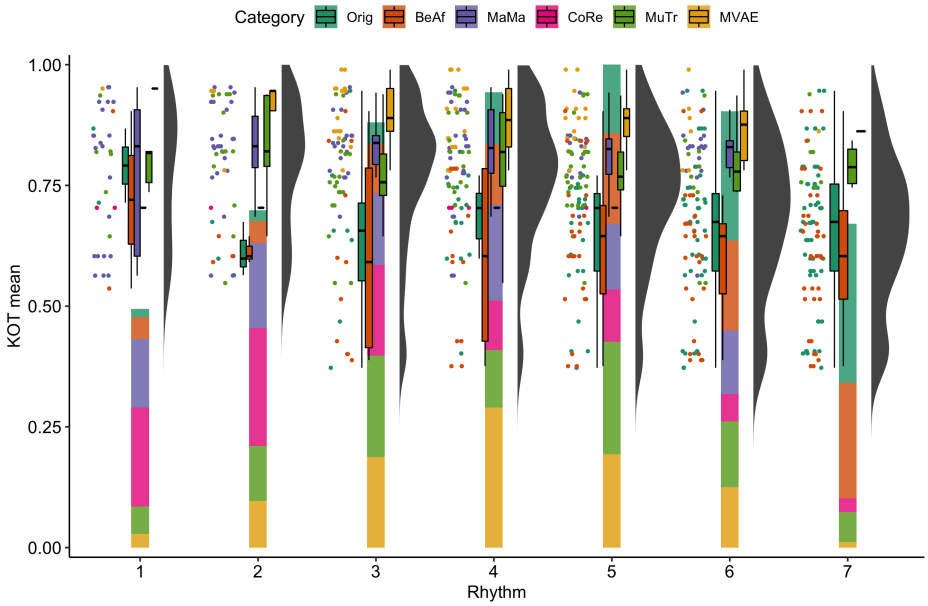


Figure B.48: KOT mean against Rhythm ratings.

B.1.9 KOT variance

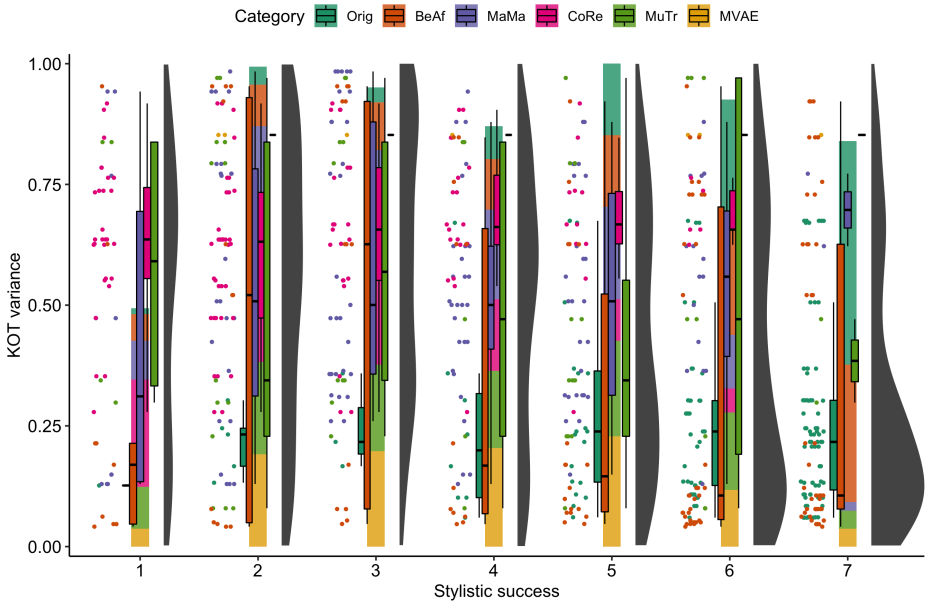


Figure B.49: KOT variance against Stylistic success ratings.

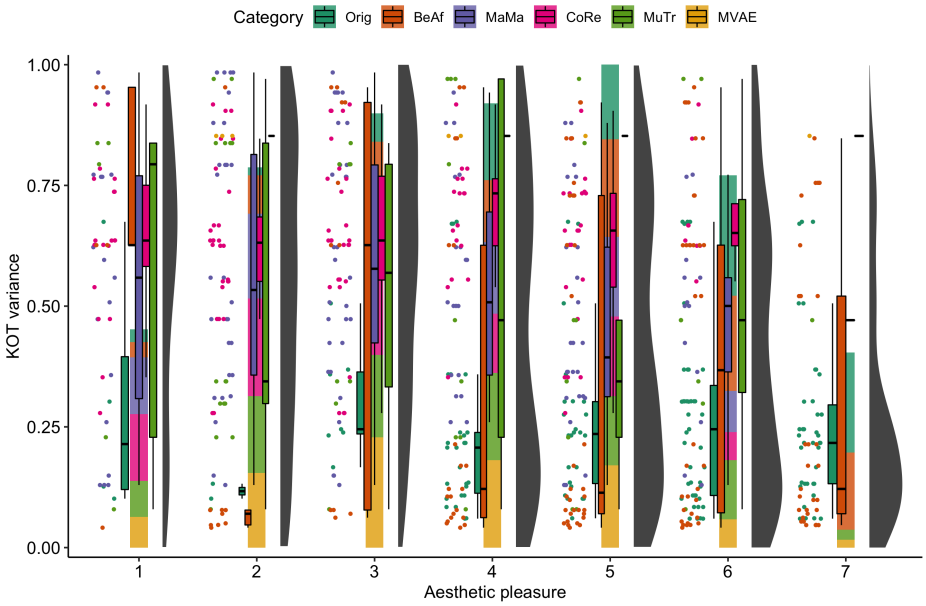


Figure B.50: KOT variance against Aesthetic pleasure ratings.

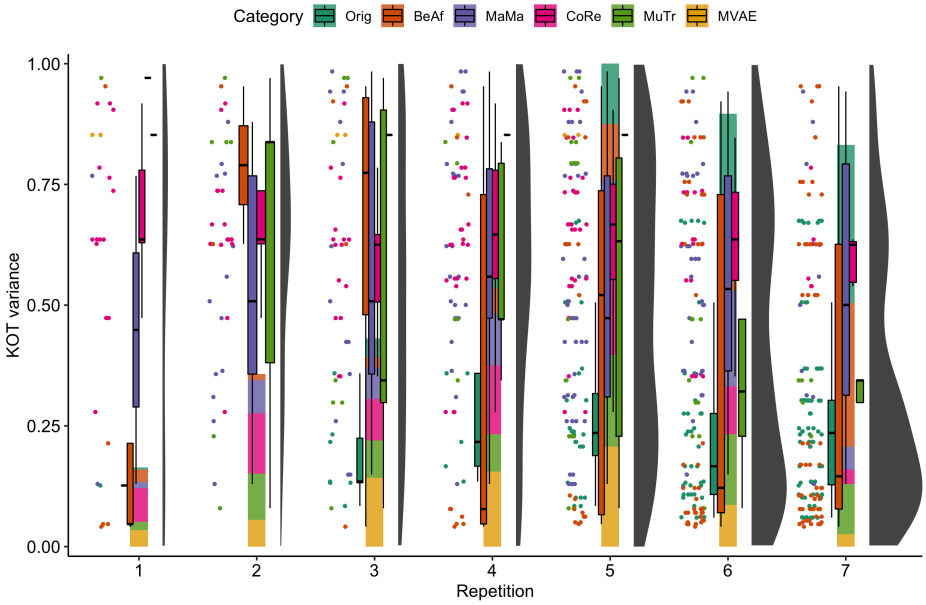


Figure B.51: KOT variance against Repetition ratings.

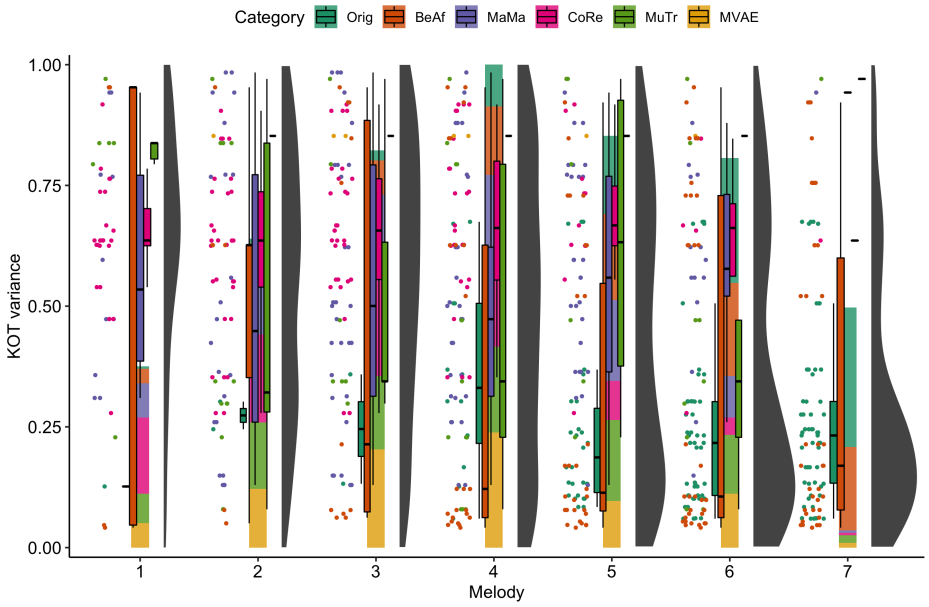


Figure B.52: KOT variance against Melody ratings.

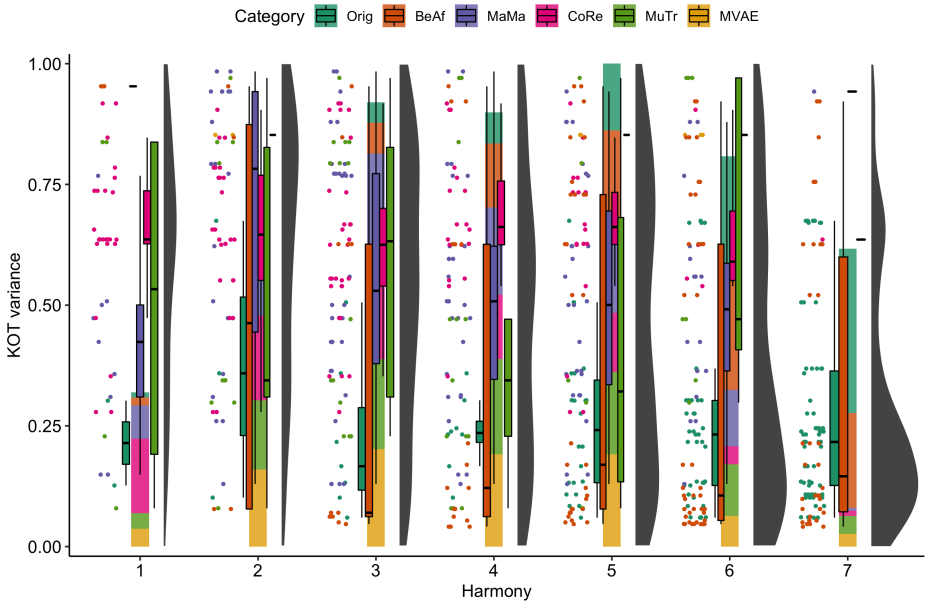


Figure B.53: KOT variance against Harmony ratings.

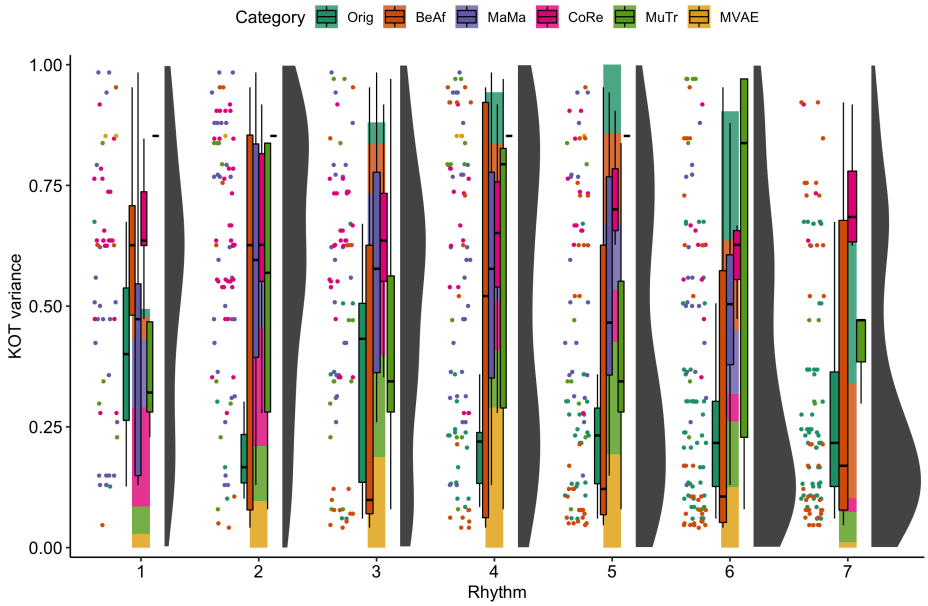


Figure B.54: KOT variance against Rhythm ratings.

B.1.10 KDT mean

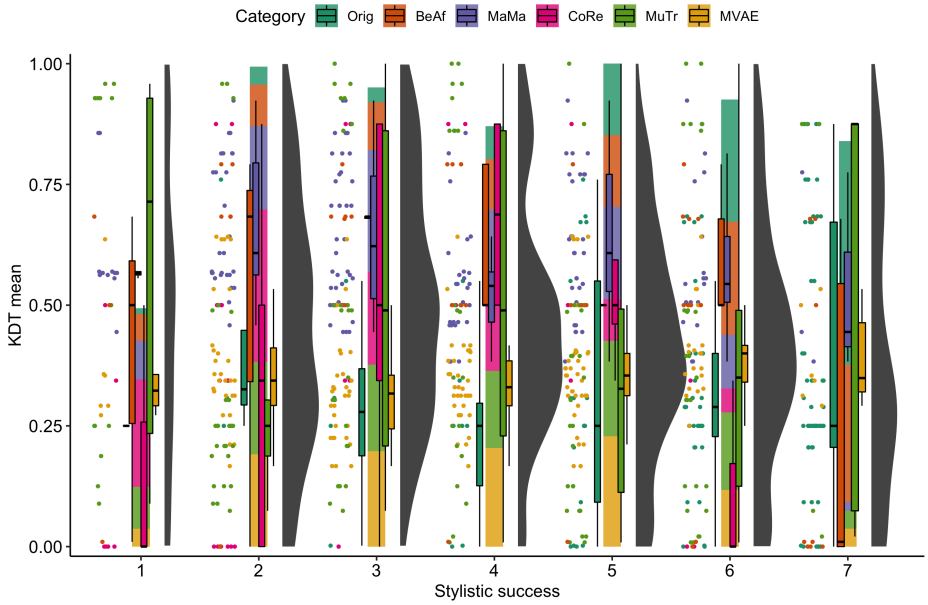


Figure B.55: KDT mean against Stylistic success ratings.

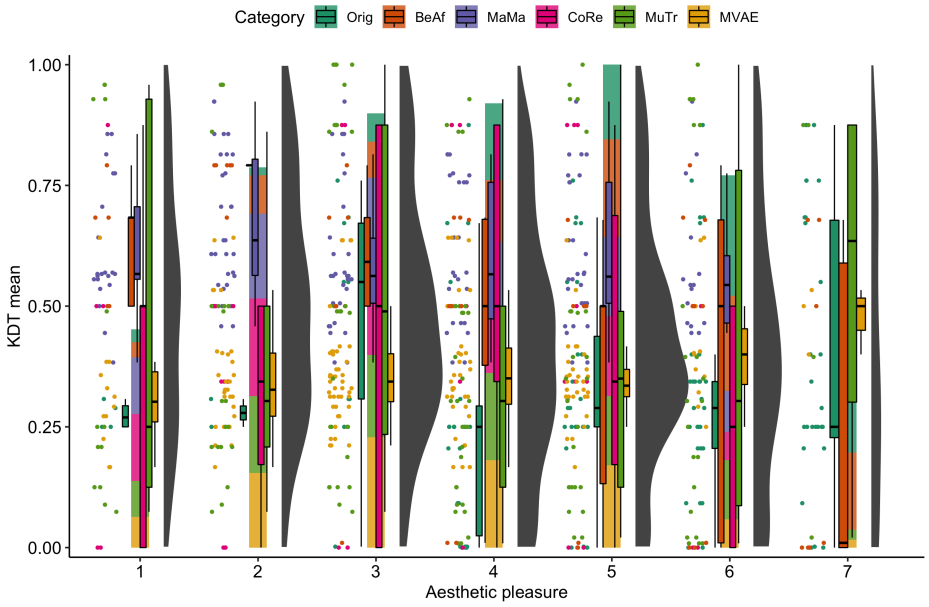


Figure B.56: KDT mean against Aesthetic pleasure ratings.

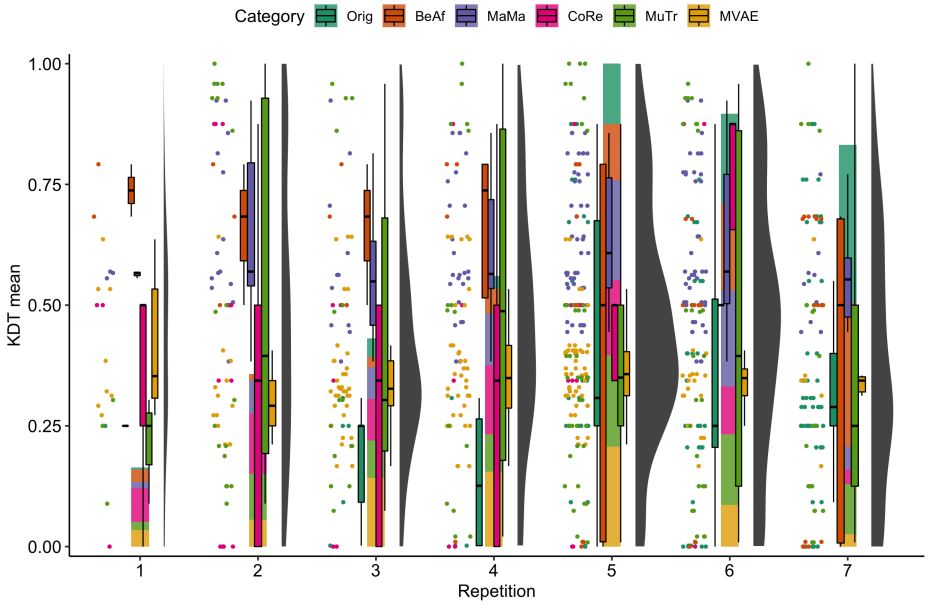


Figure B.57: KDT mean against Repetition ratings.

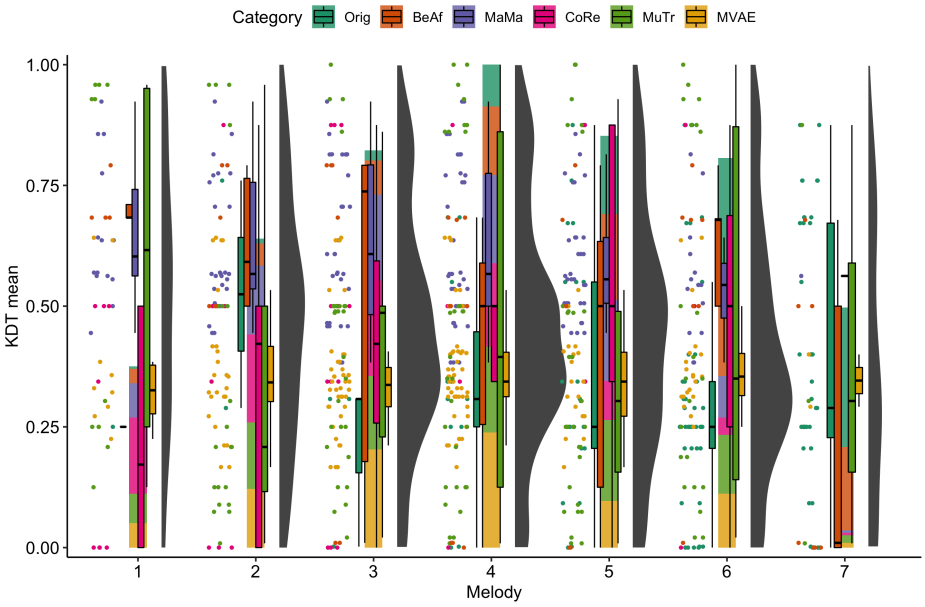


Figure B.58: KDT mean against Melody ratings.

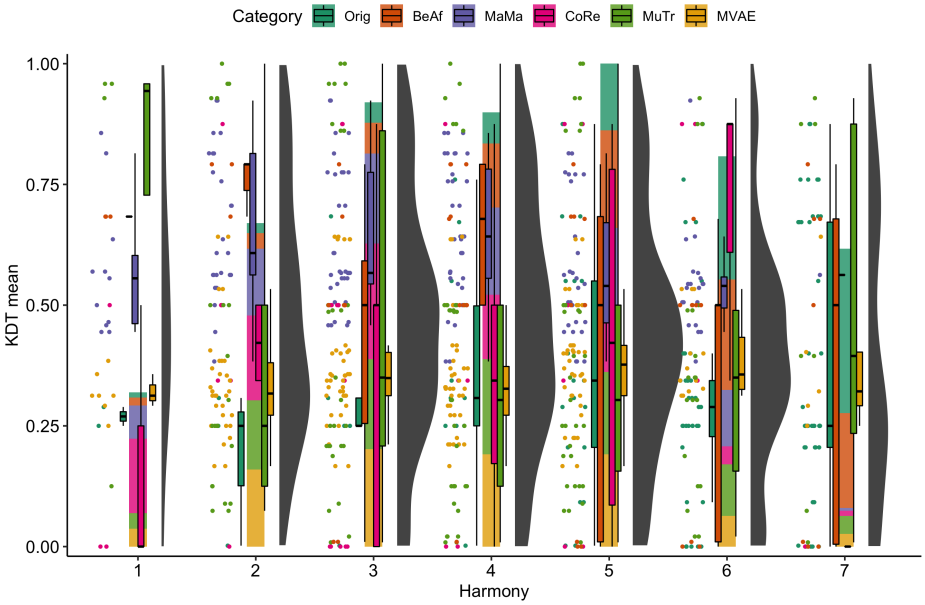


Figure B.59: KDT mean against Harmony ratings.

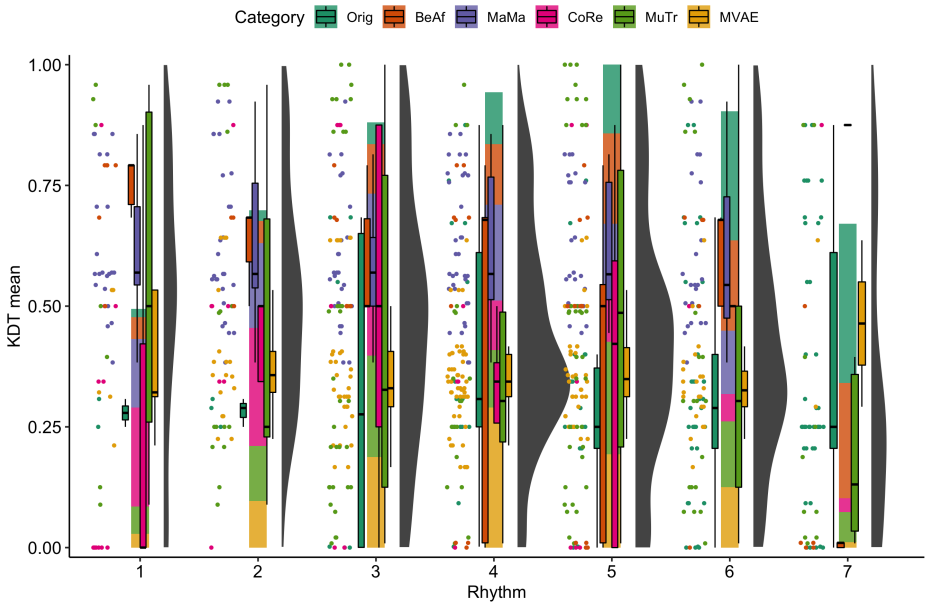


Figure B.60: KDT mean against Rhythm ratings.

B.1.11 KDT variance

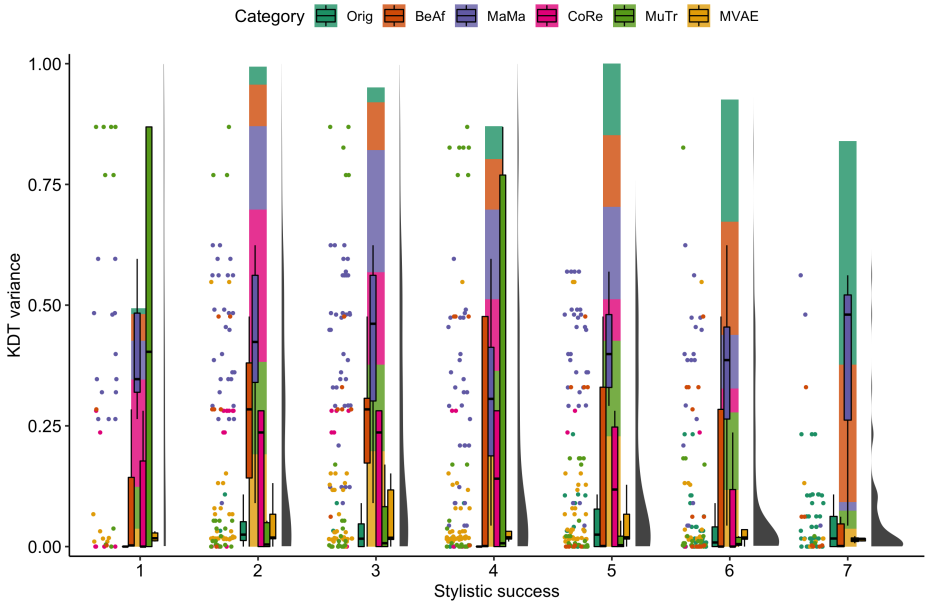


Figure B.61: KDT variance against Stylistic success ratings.

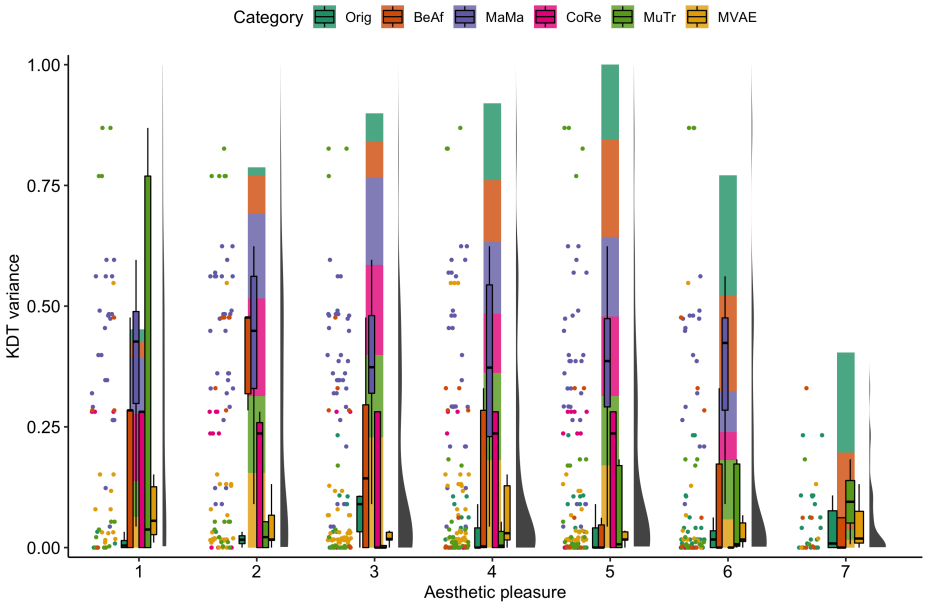


Figure B.62: KDT variance against Aesthetic pleasure ratings.

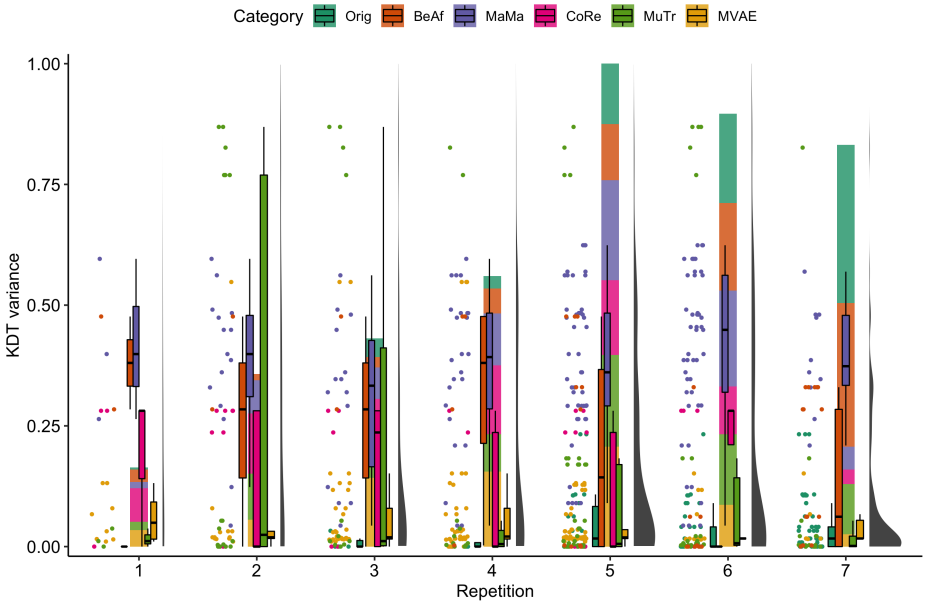


Figure B.63: KDT variance against Repetition ratings.

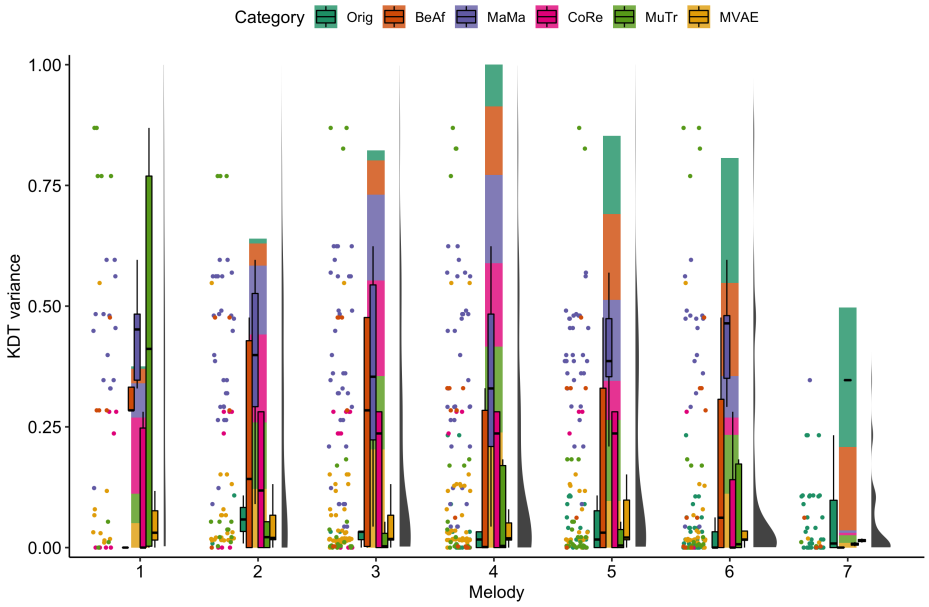


Figure B.64: KDT variance against Melody ratings.

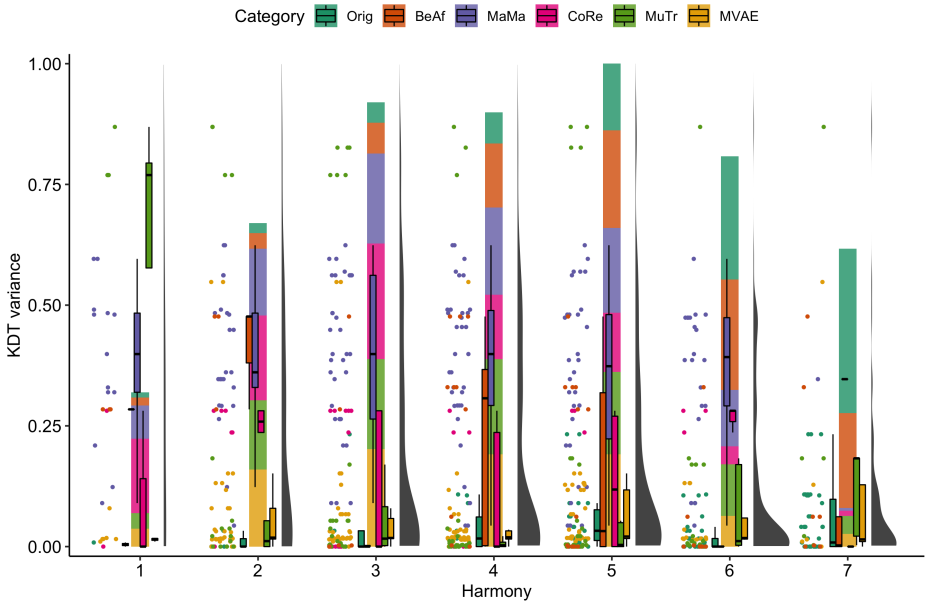


Figure B.65: KDT variance against Harmony ratings.

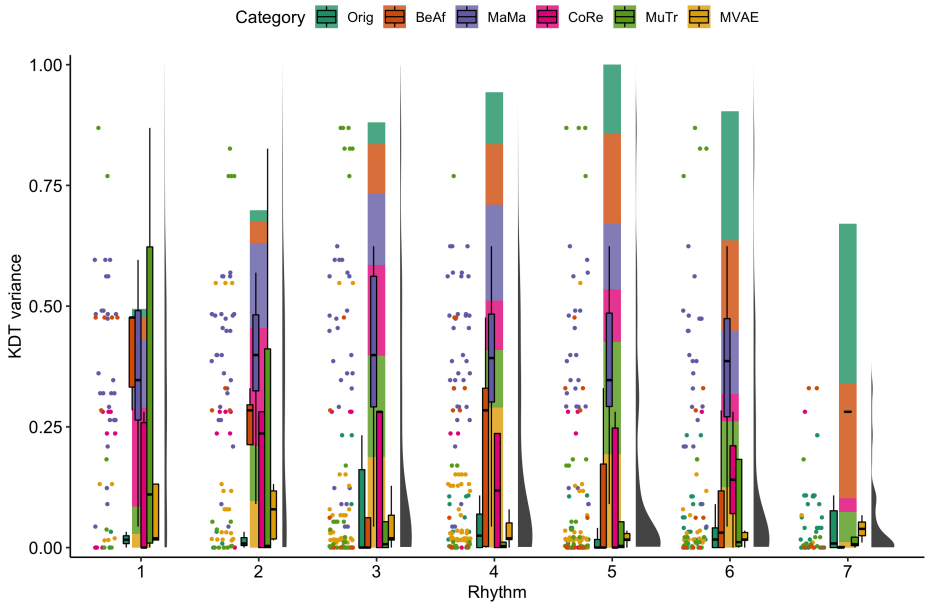


Figure B.66: KDT variance against Rhythm ratings.

B.1.12 Jitter mean

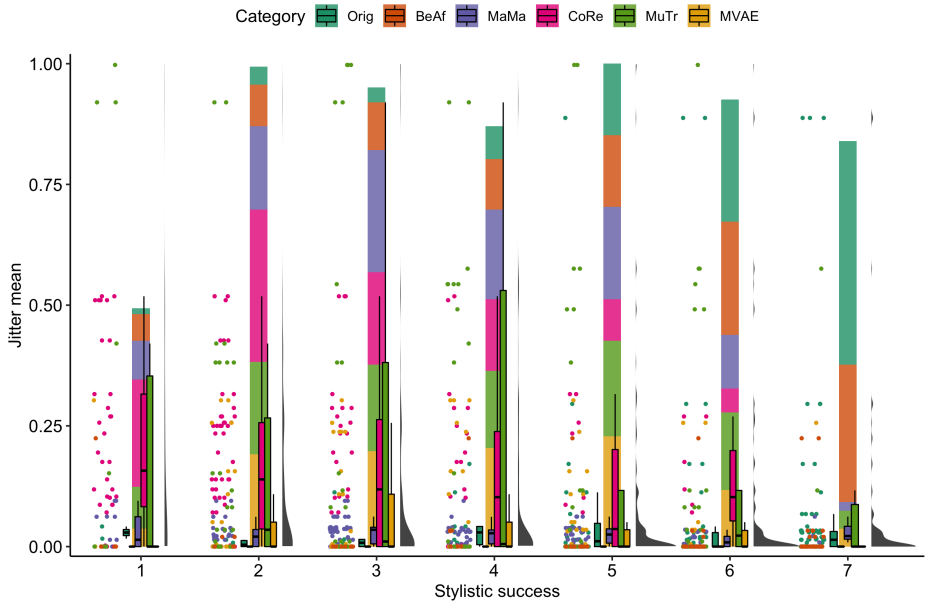


Figure B.67: Jitter mean against Stylistic success ratings.

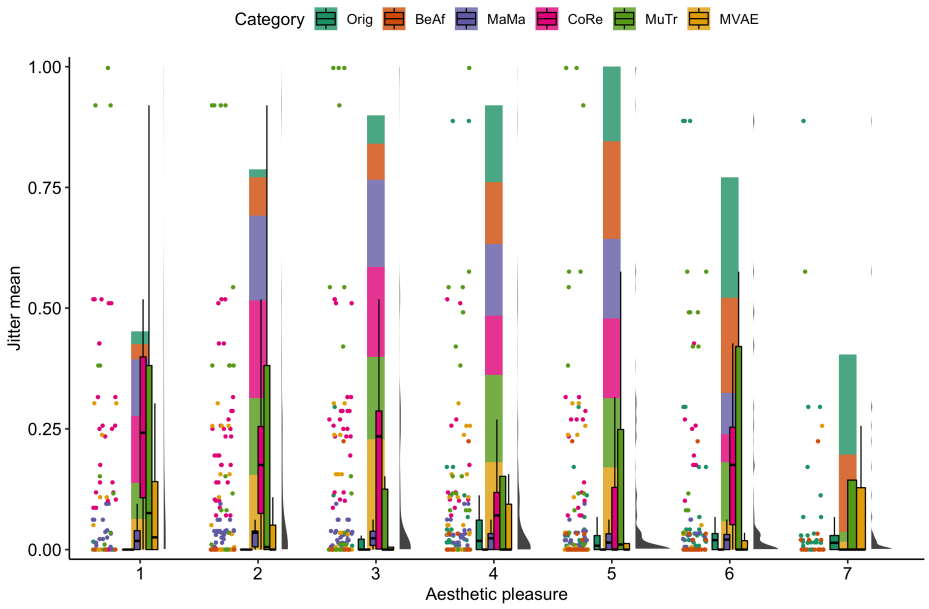


Figure B.68: Jitter mean against Aesthetic pleasure ratings.

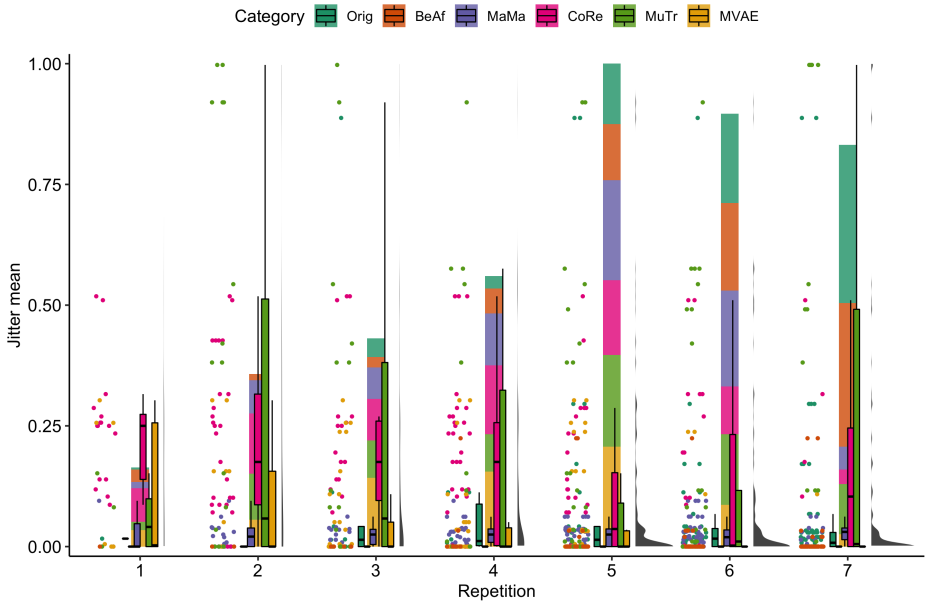


Figure B.69: Jitter mean against Repetition ratings.

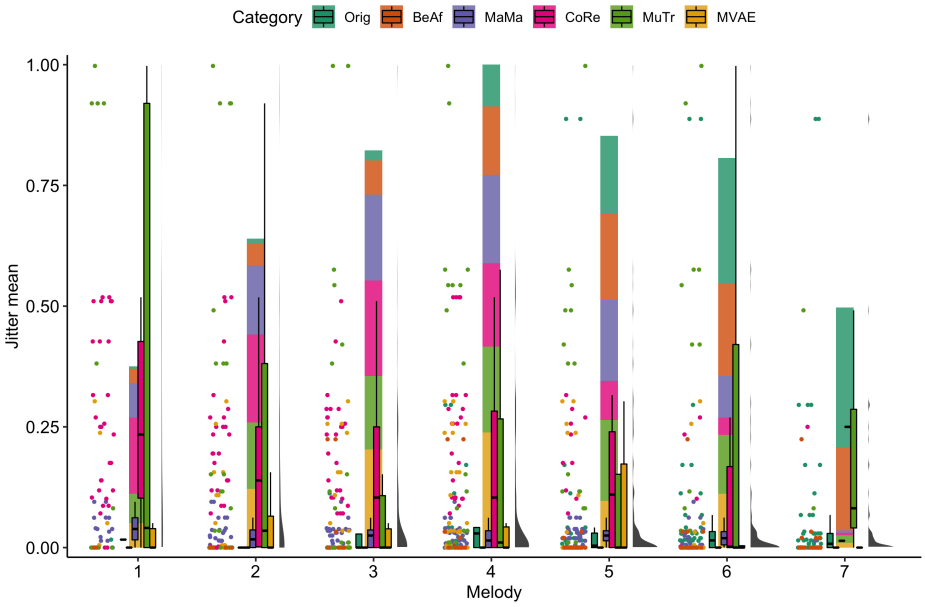


Figure B.70: Jitter mean against Melody ratings.

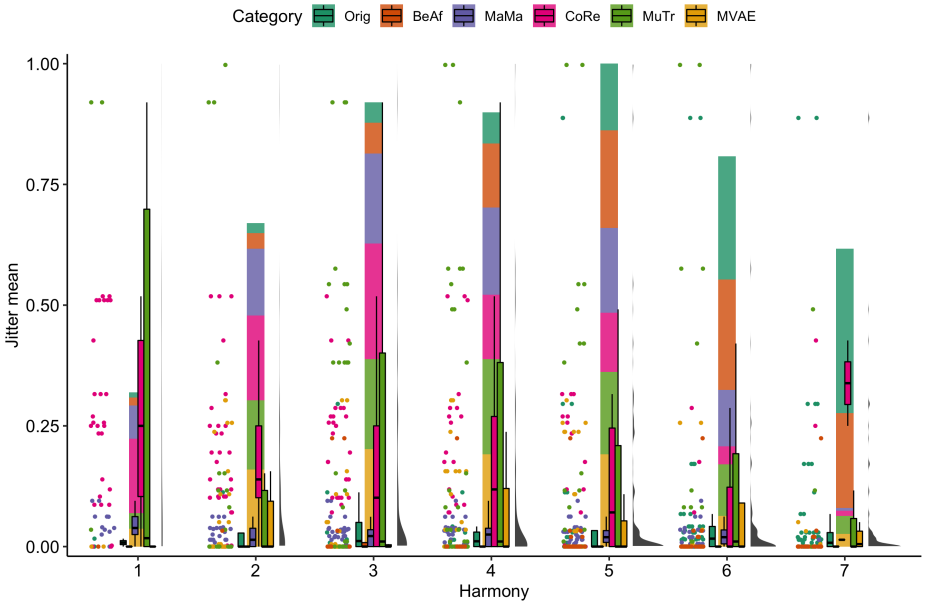


Figure B.71: Jitter mean against Harmony ratings.

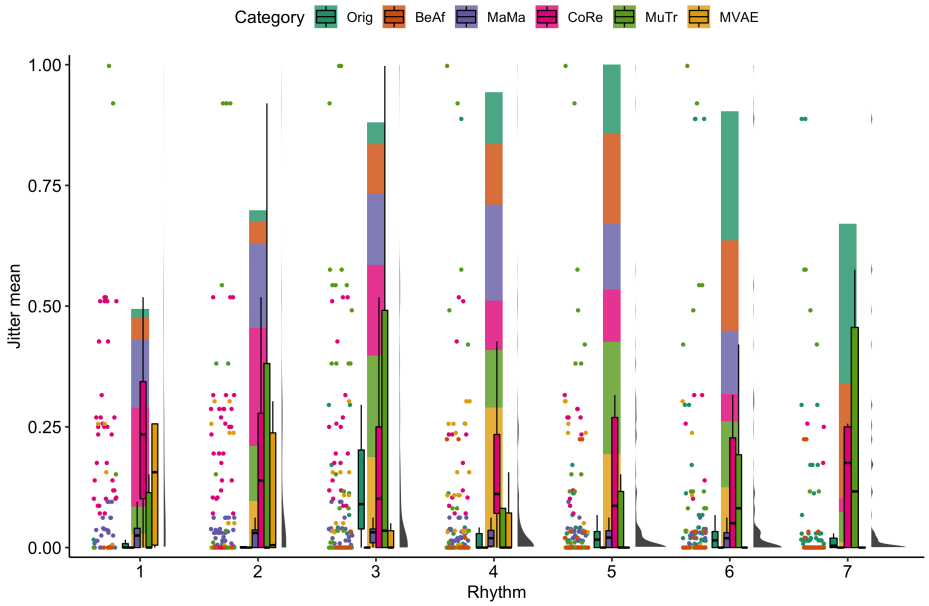


Figure B.72: Jitter mean against Rhythm ratings.

B.1.13 Jitter variance

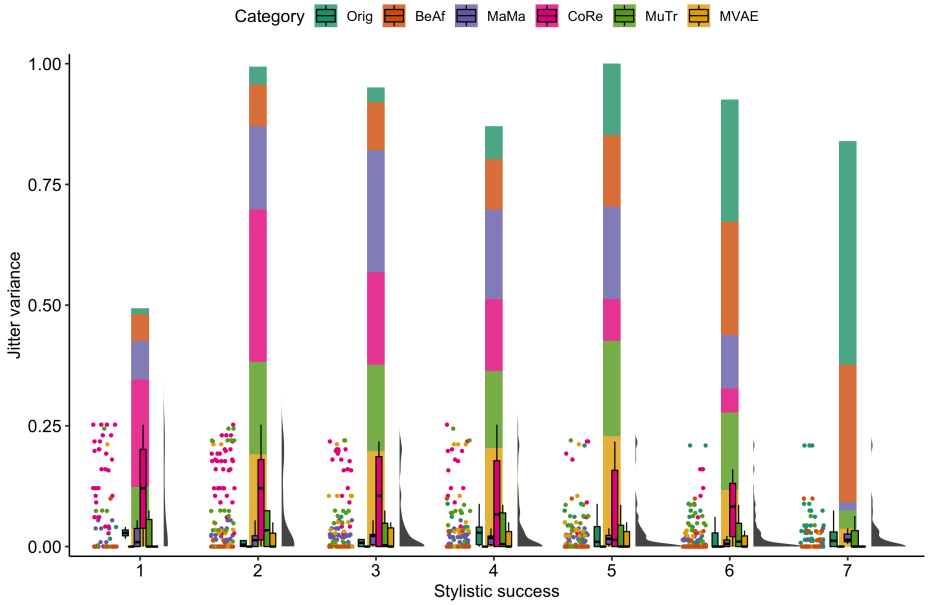


Figure B.73: Jitter variance against Stylistic success ratings.

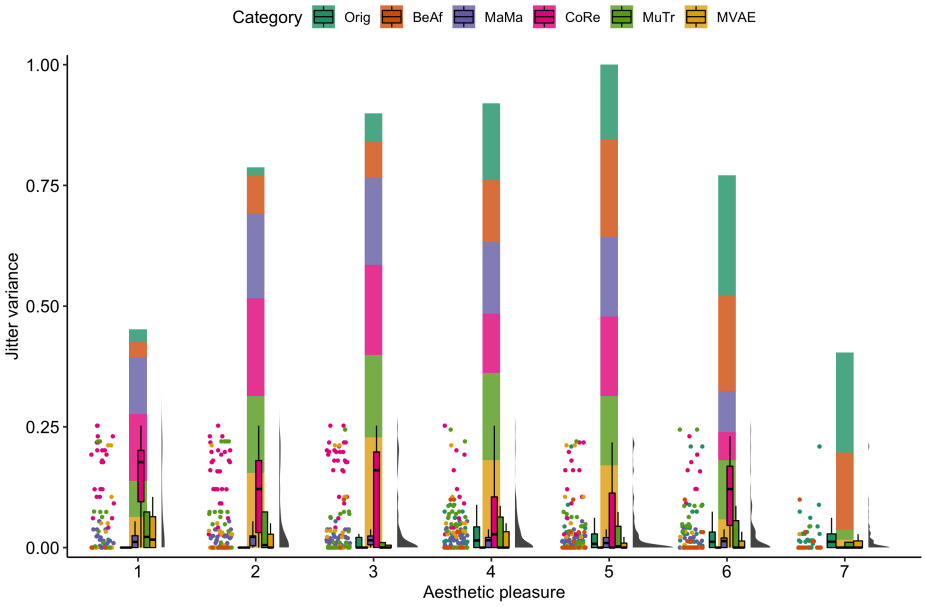


Figure B.74: Jitter variance against Aesthetic pleasure ratings.

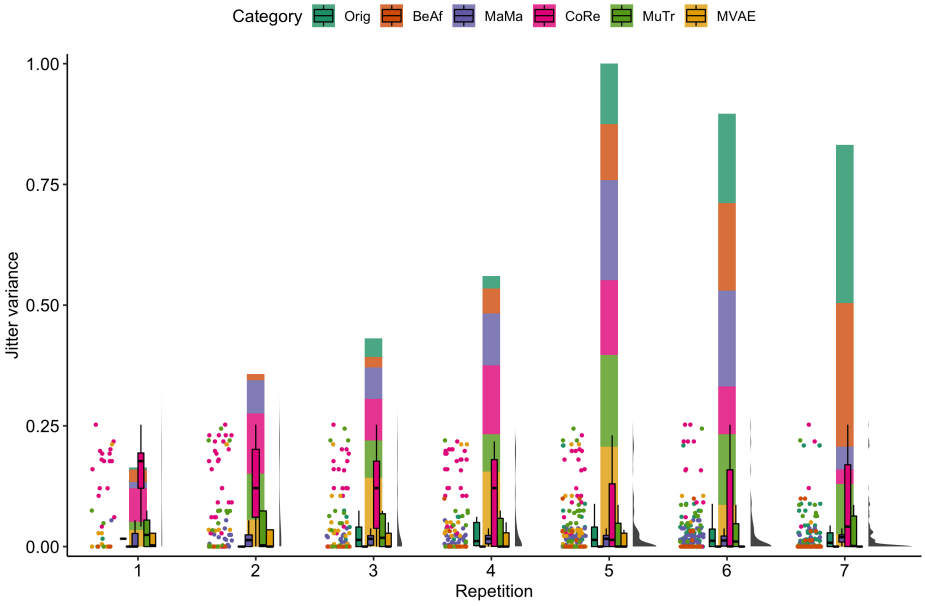


Figure B.75: Jitter variance against Repetition ratings.

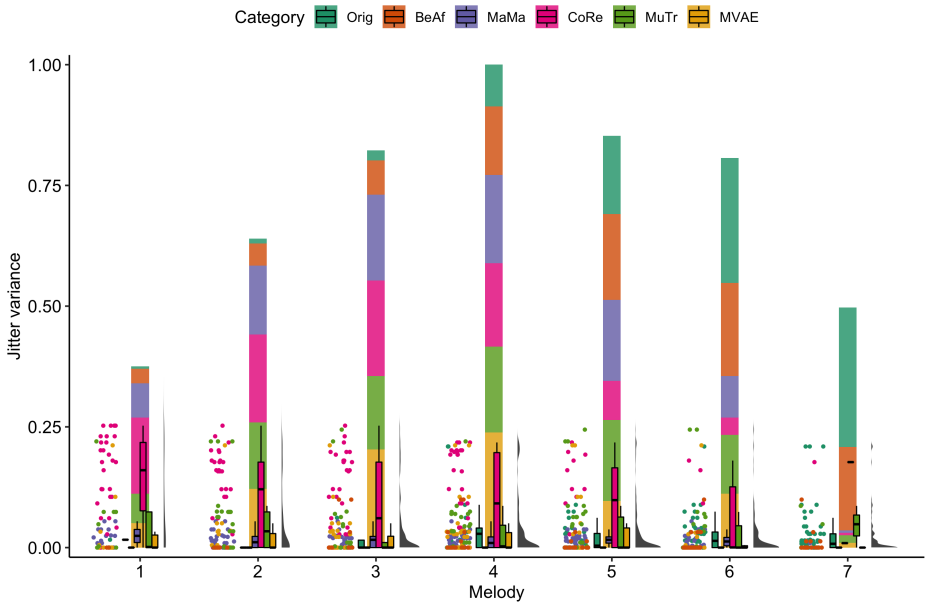


Figure B.76: Jitter variance against Melody ratings.

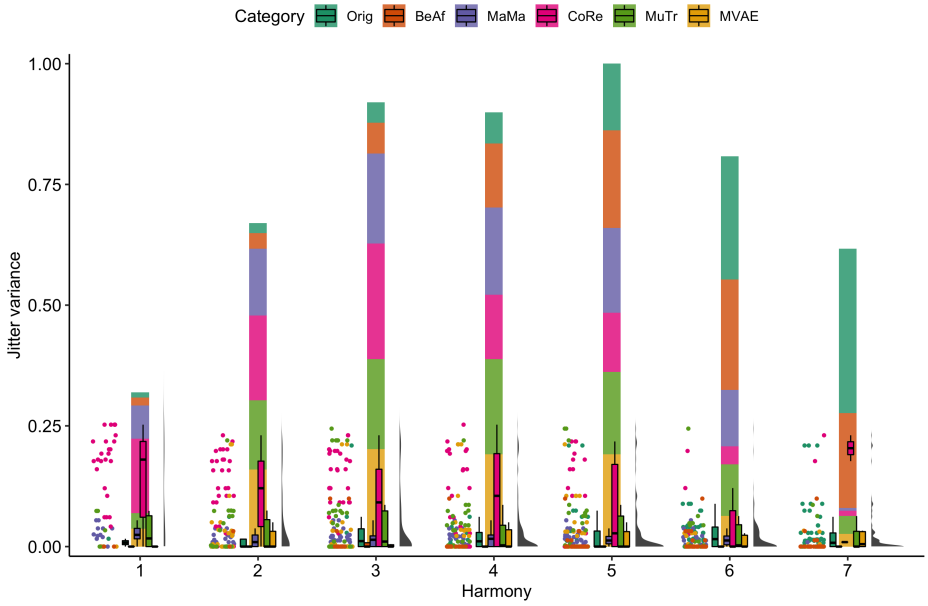


Figure B.77: Jitter variance against Harmony ratings.

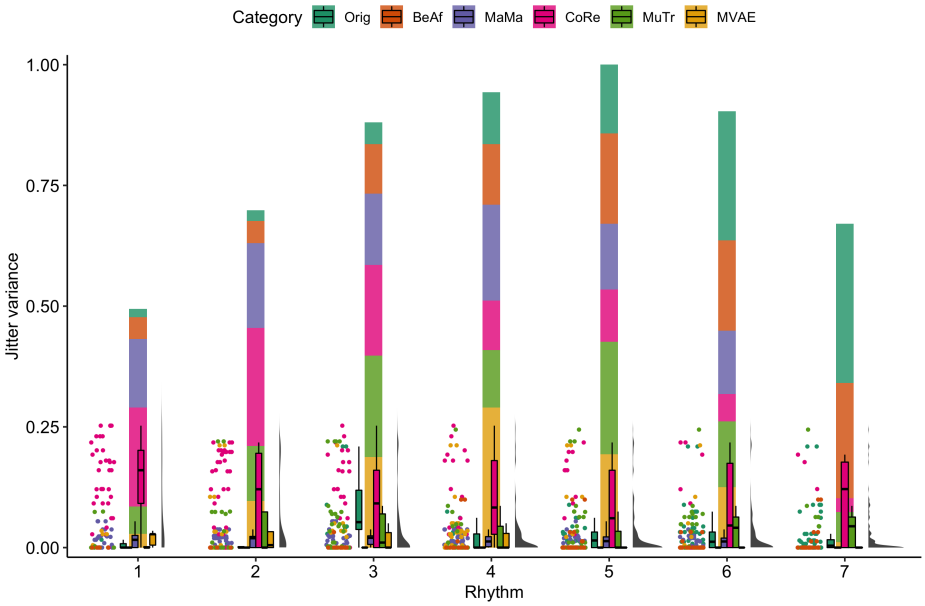


Figure B.78: Jitter variance against Rhythm ratings.

B.2 Classical string quartets

B.2.1 Translational Complexity

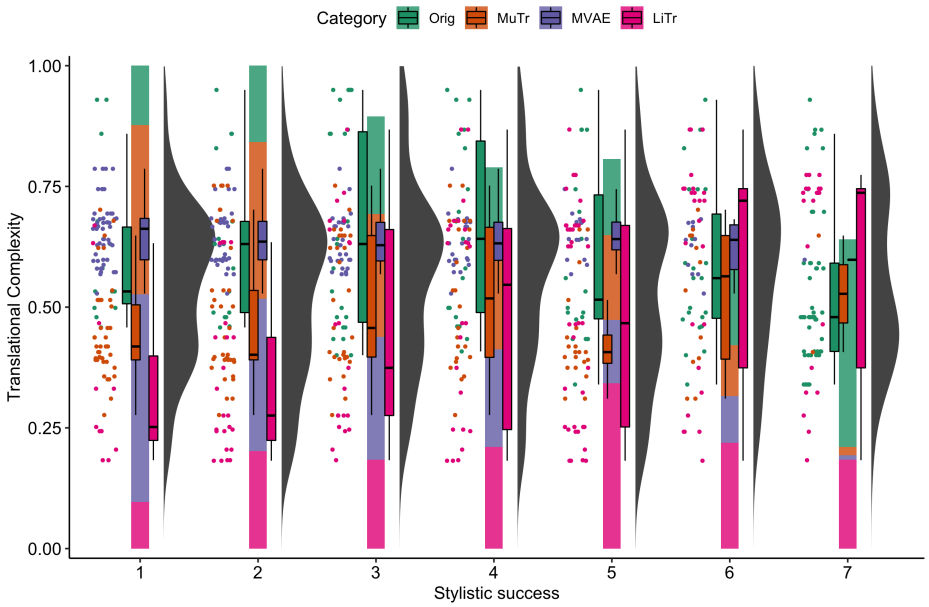


Figure B.79: Translational Complexity against Stylistic success ratings.

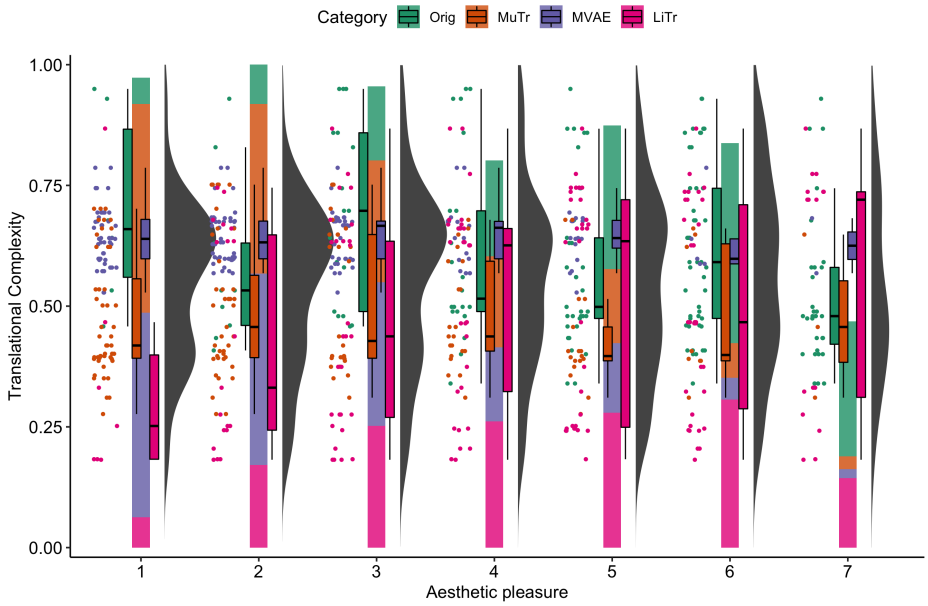


Figure B.80: Translational Complexity against Aesthetic pleasure ratings.

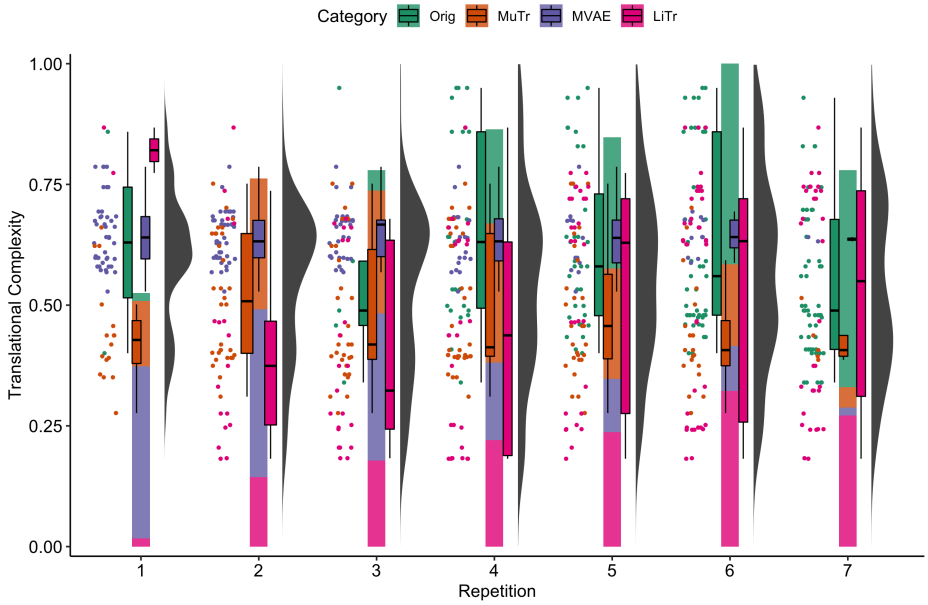


Figure B.81: Translational Complexity against Repetition ratings.

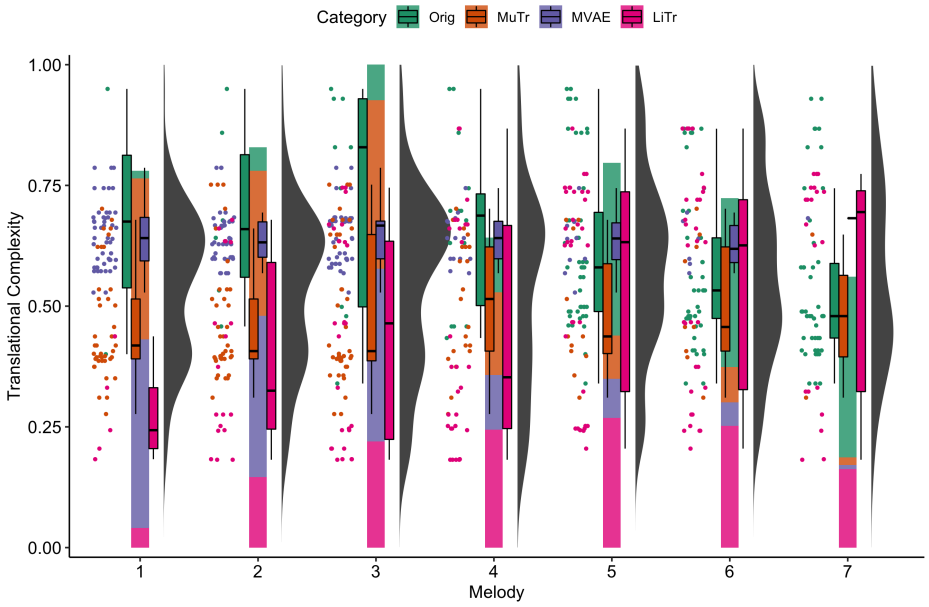


Figure B.82: Translational Complexity against Melody ratings.

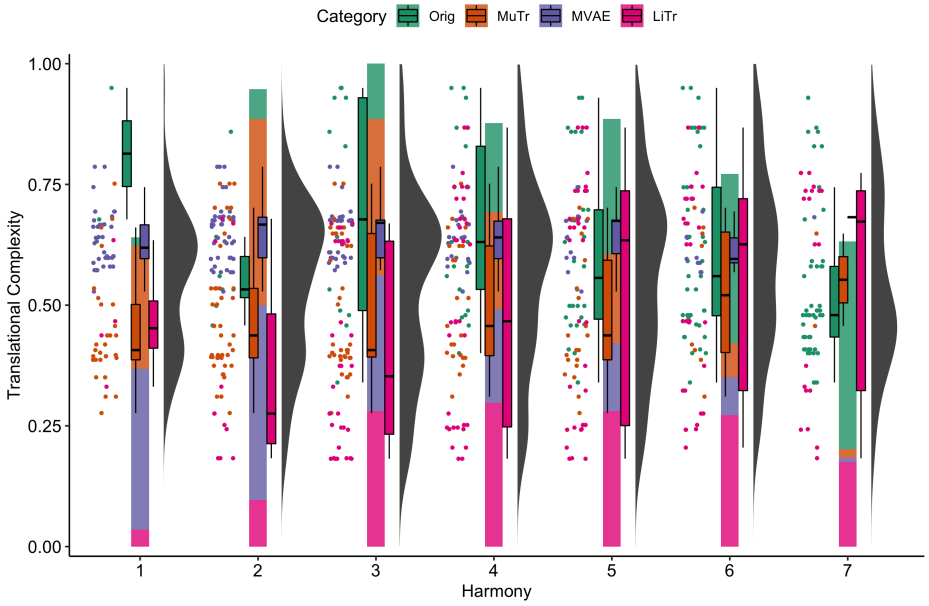


Figure B.83: Translational Complexity against Harmony ratings.

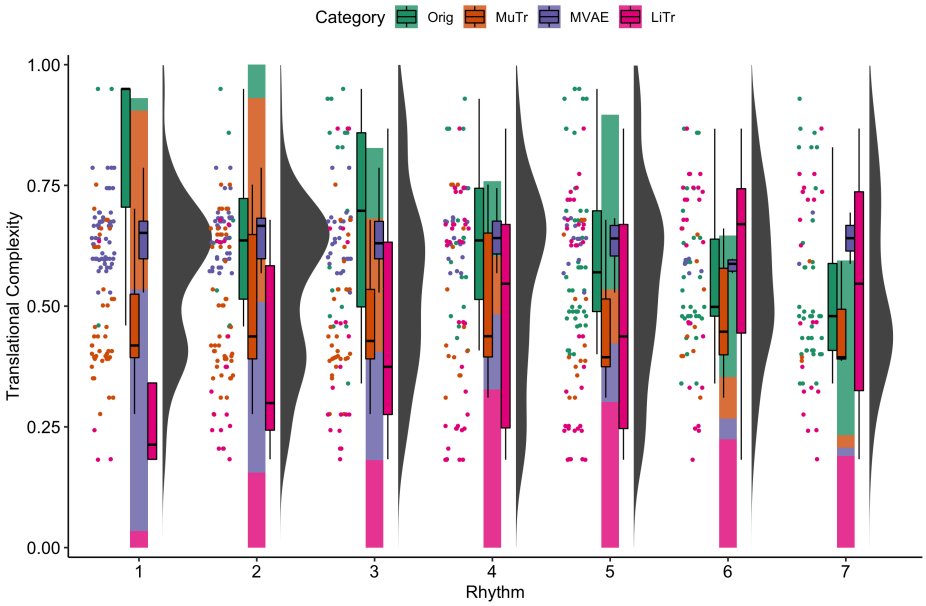


Figure B.84: Translational Complexity against Rhythm ratings.

B.2.2 Arc score

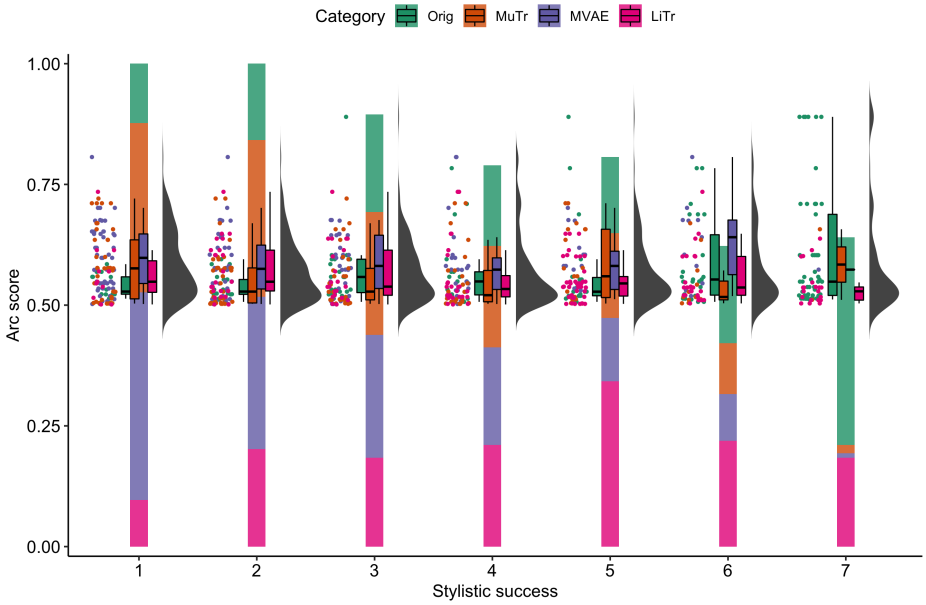


Figure B.85: Arc score against Stylistic success ratings.

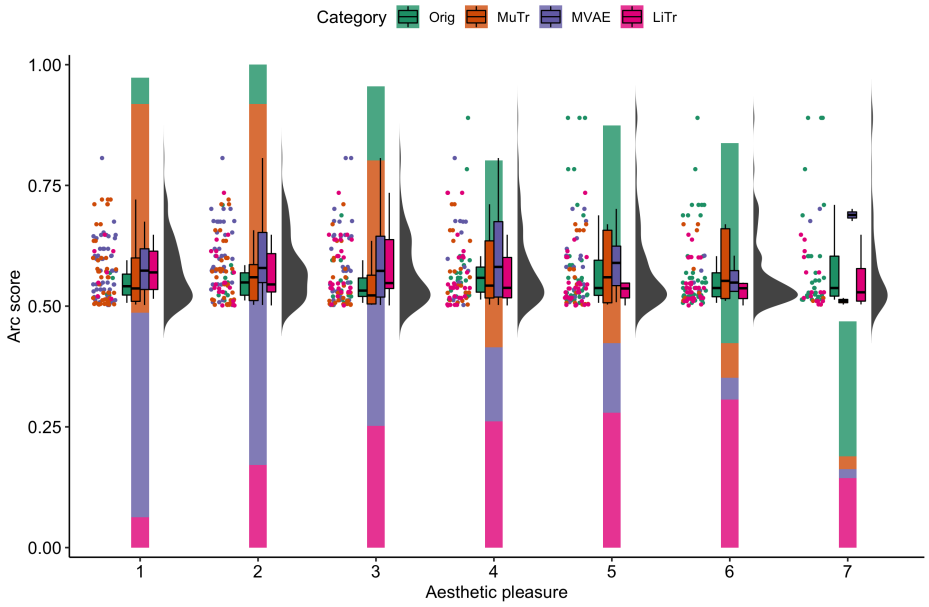


Figure B.86: Arc score against Aesthetic pleasure ratings.

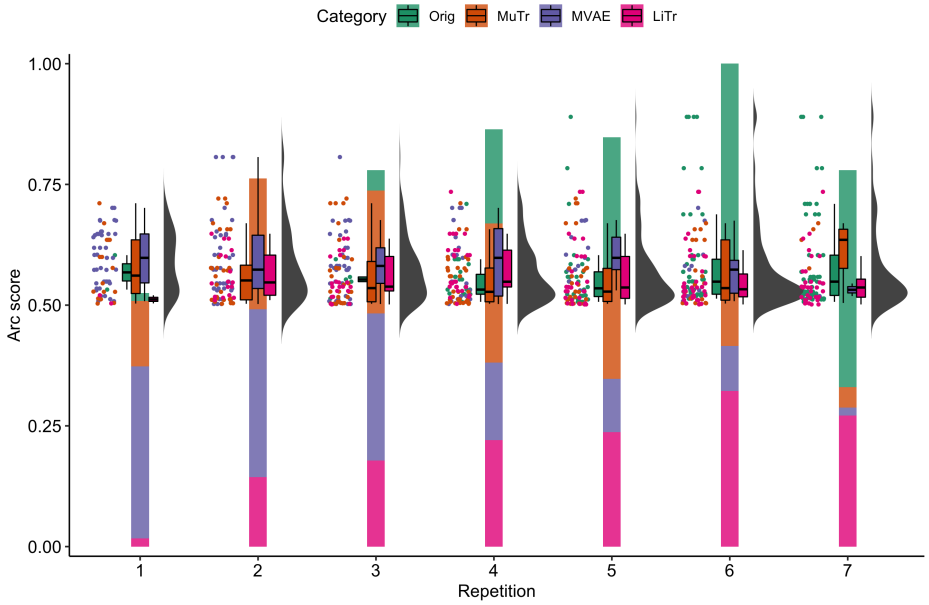


Figure B.87: Arc score against Repetition ratings.

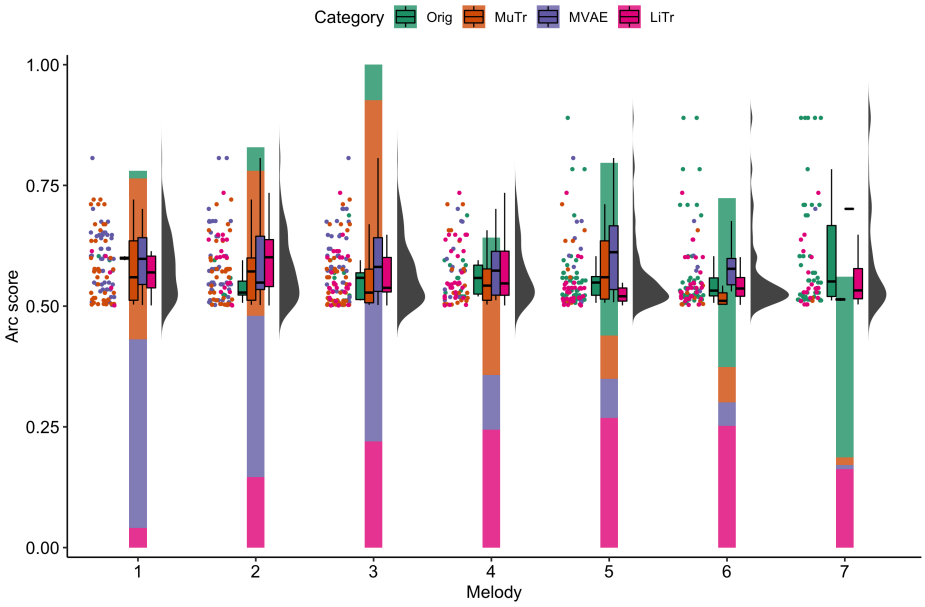


Figure B.88: Arc score against Melody ratings.

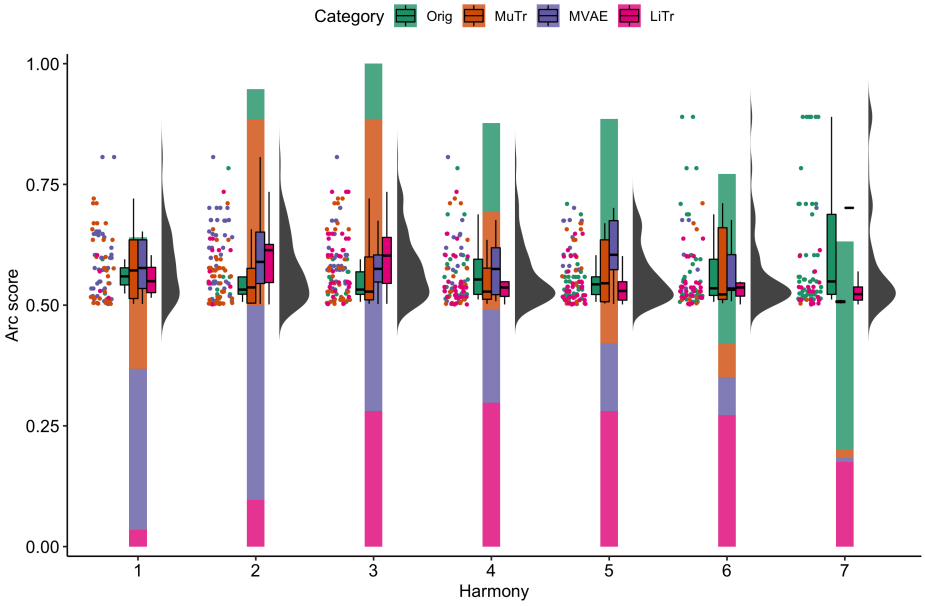


Figure B.89: Arc score against Harmony ratings.

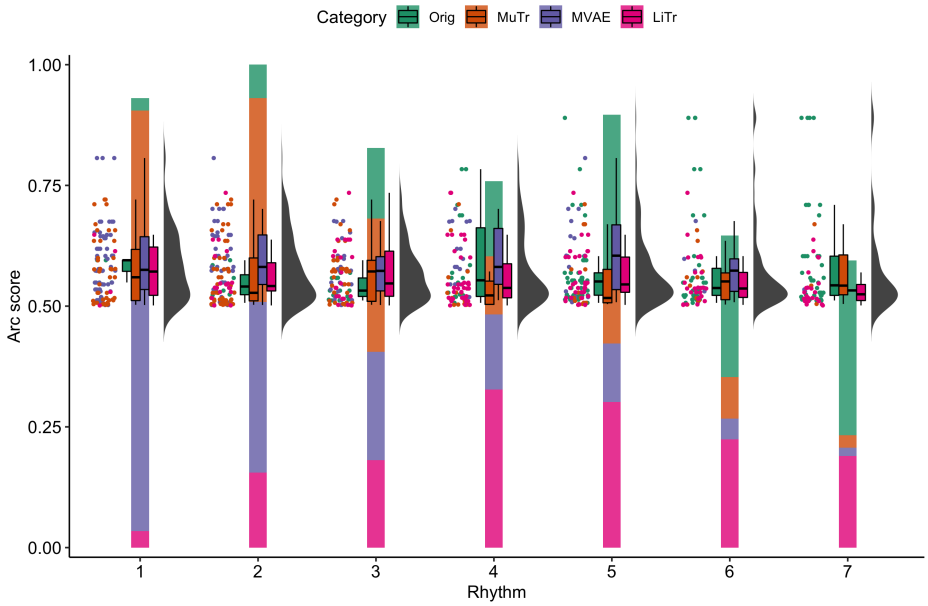


Figure B.90: Arc score against Rhythm ratings.

B.2.3 Tonal ambiguity

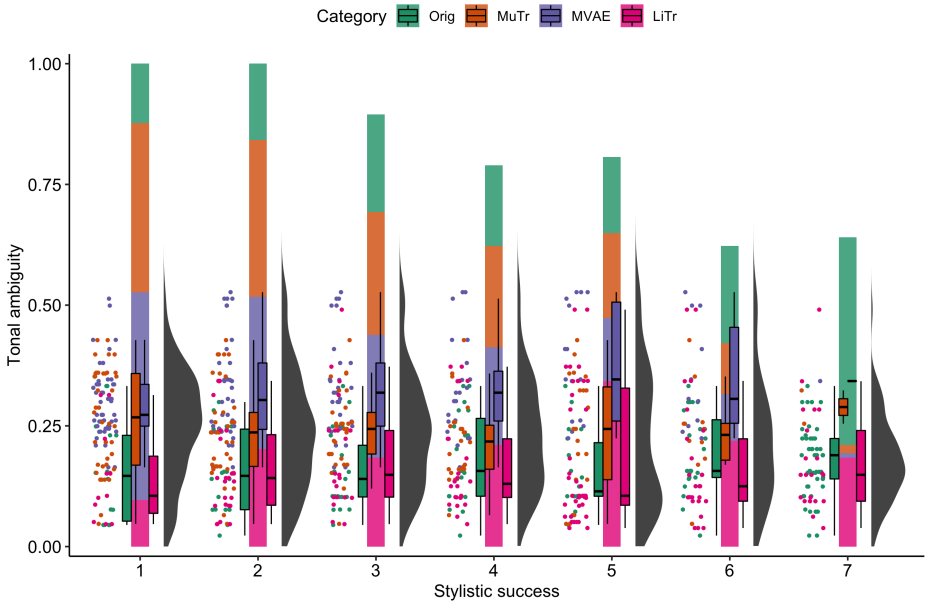


Figure B.91: Tonal ambiguity against Stylistic success ratings.

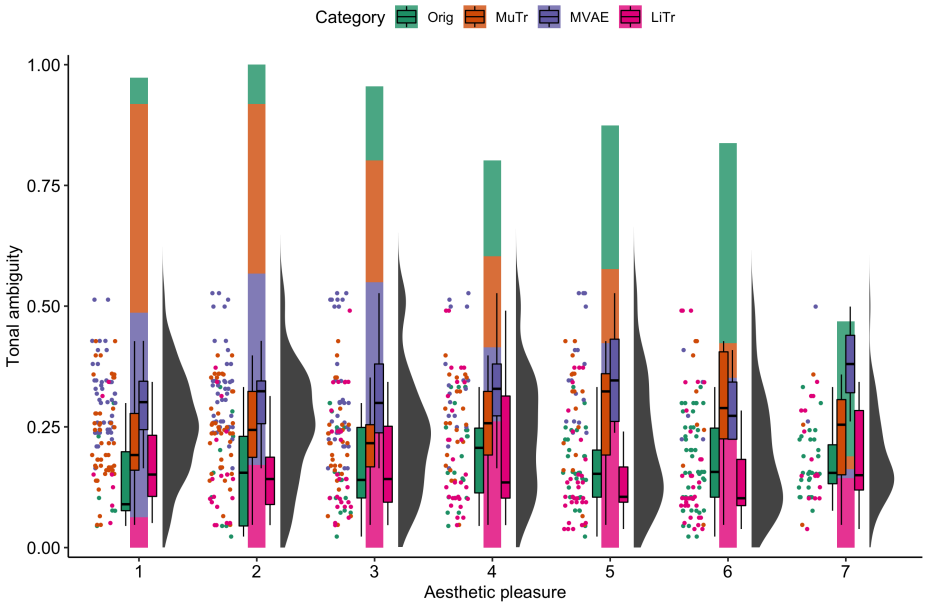


Figure B.92: Tonal ambiguity against Aesthetic pleasure ratings.

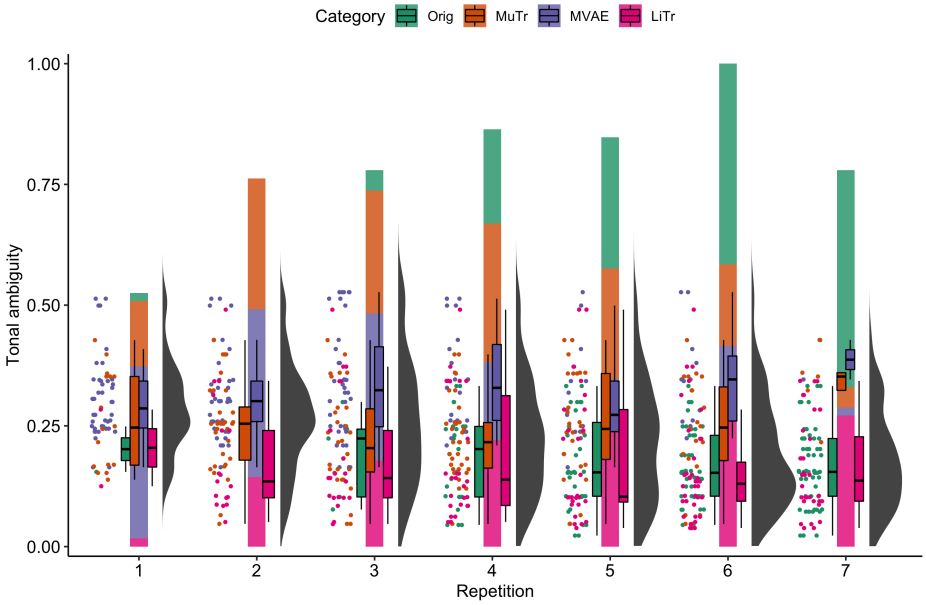


Figure B.93: Tonal ambiguity against Repetition ratings.

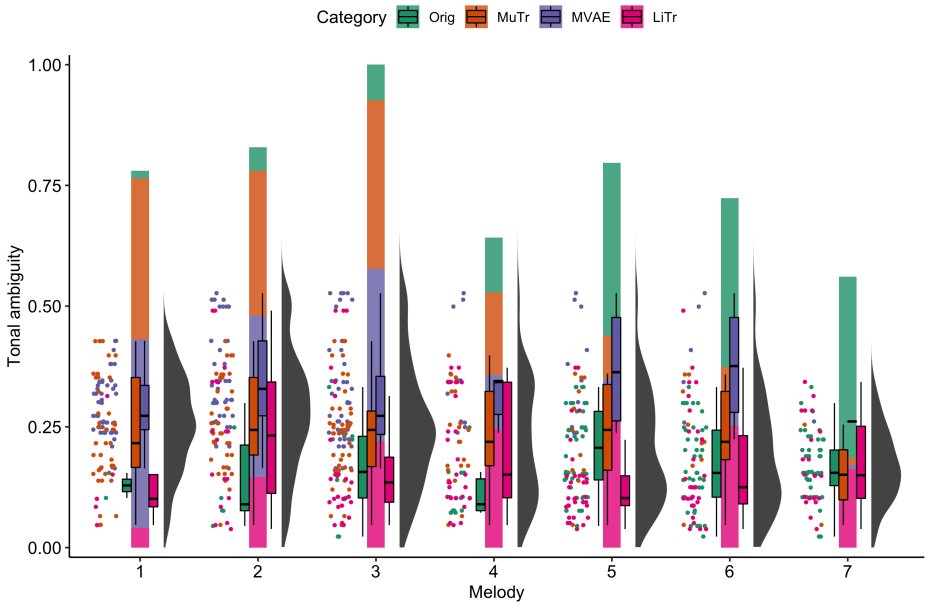


Figure B.94: Tonal ambiguity against Melody ratings.

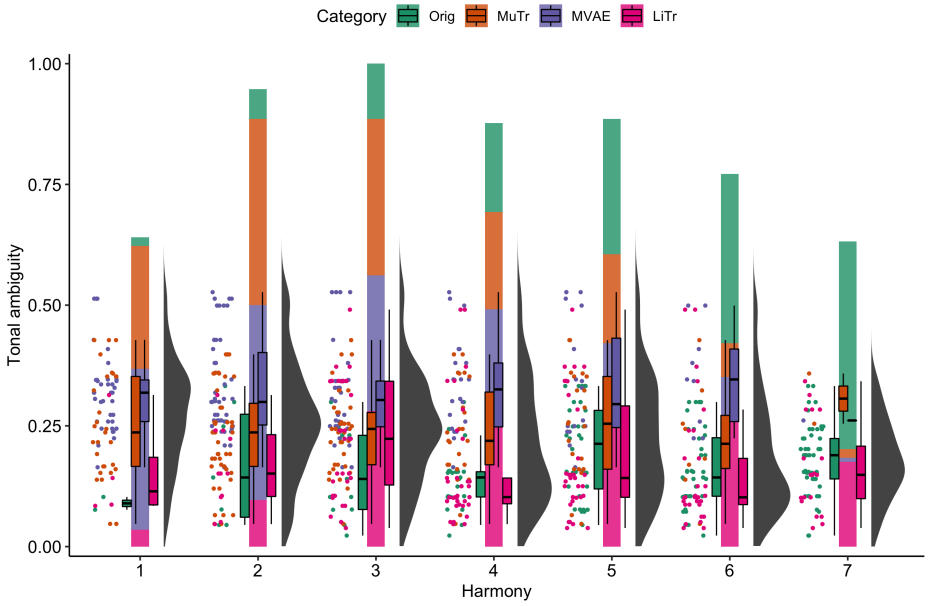


Figure B.95: Tonal ambiguity against Harmony ratings.

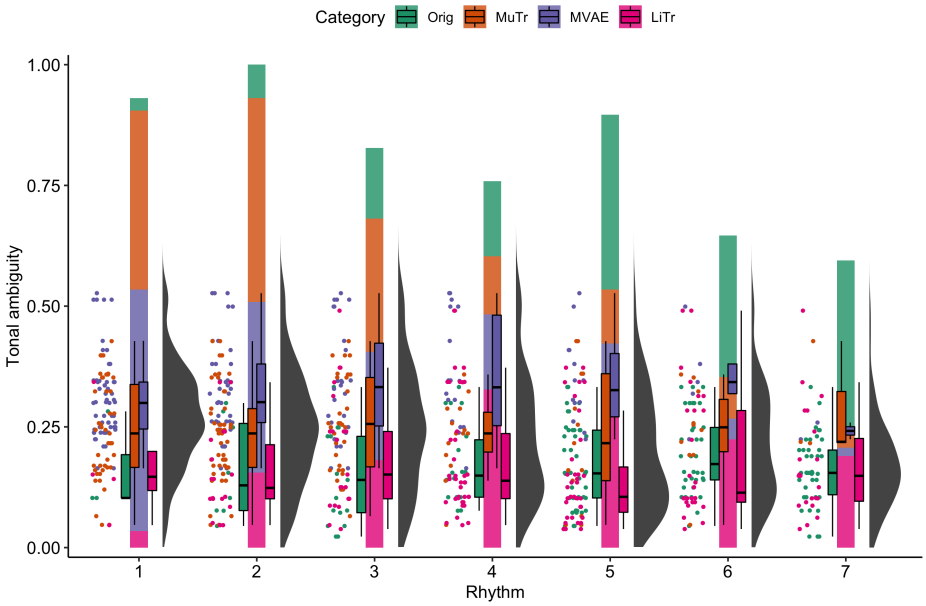


Figure B.96: Tonal ambiguity against Rhythm ratings.

B.2.4 IOI mean

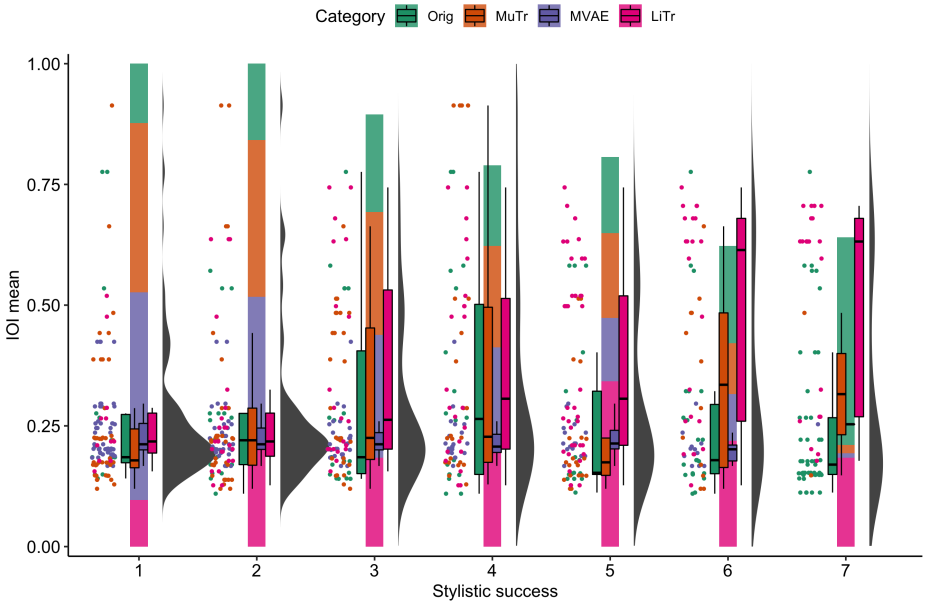


Figure B.97: IOI mean against Stylistic success ratings.

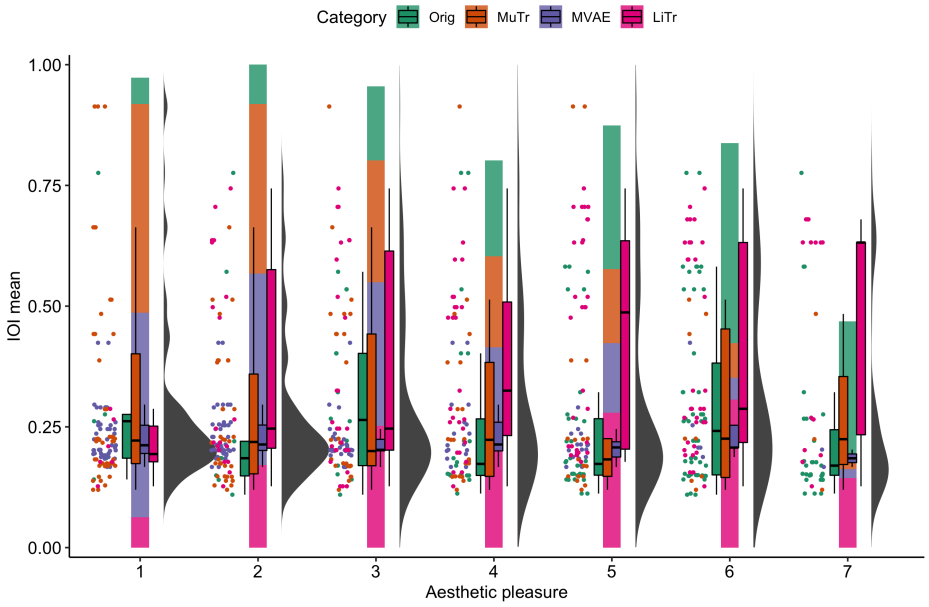


Figure B.98: IOI mean against Aesthetic pleasure ratings.

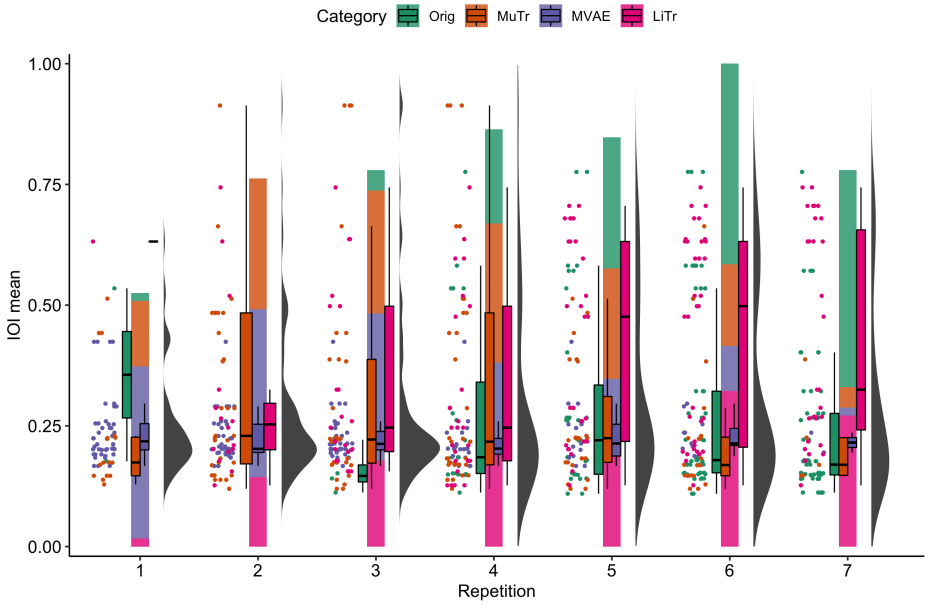


Figure B.99: IOI mean against Repetition ratings.

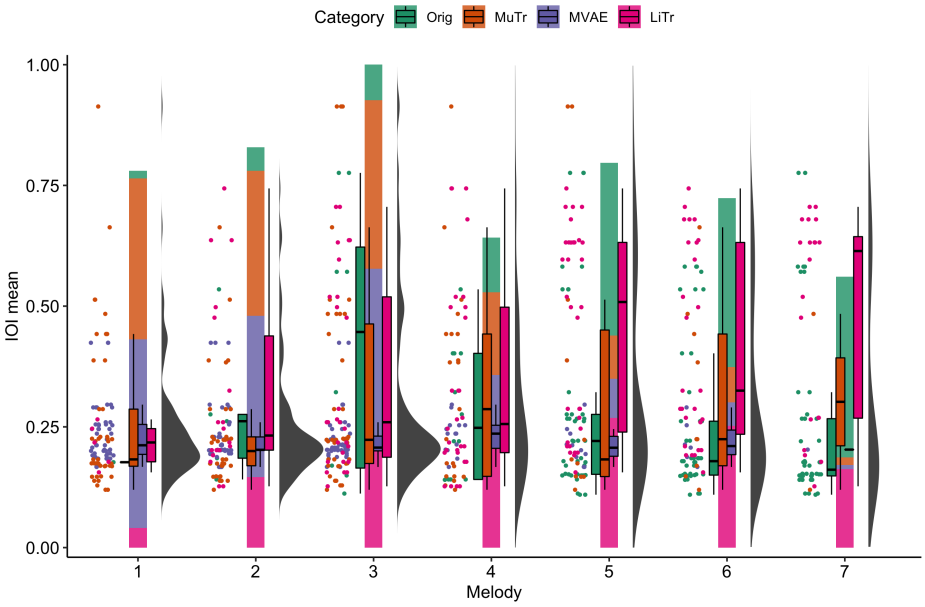


Figure B.100: IOI mean against Melody ratings.

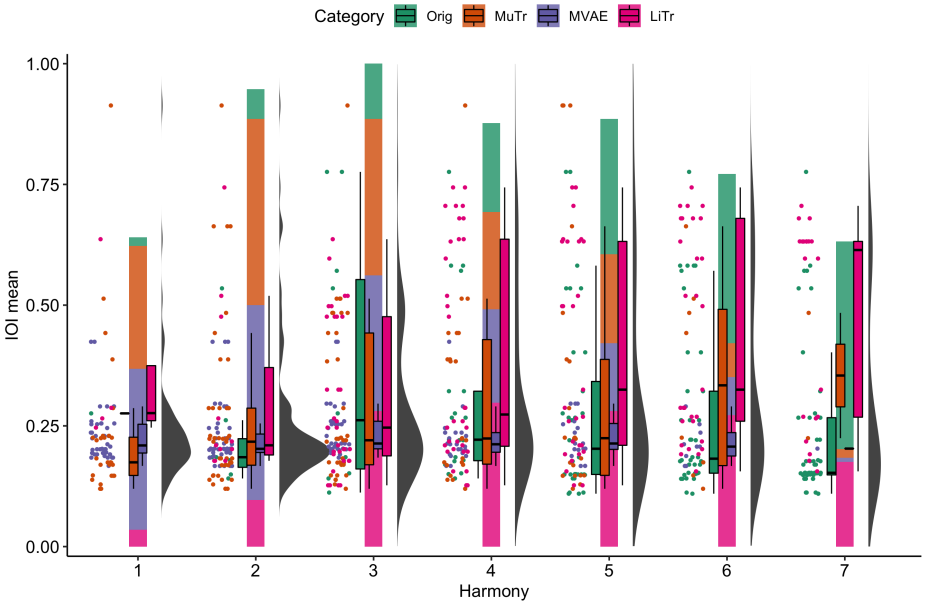


Figure B.101: IOI mean against Harmony ratings.

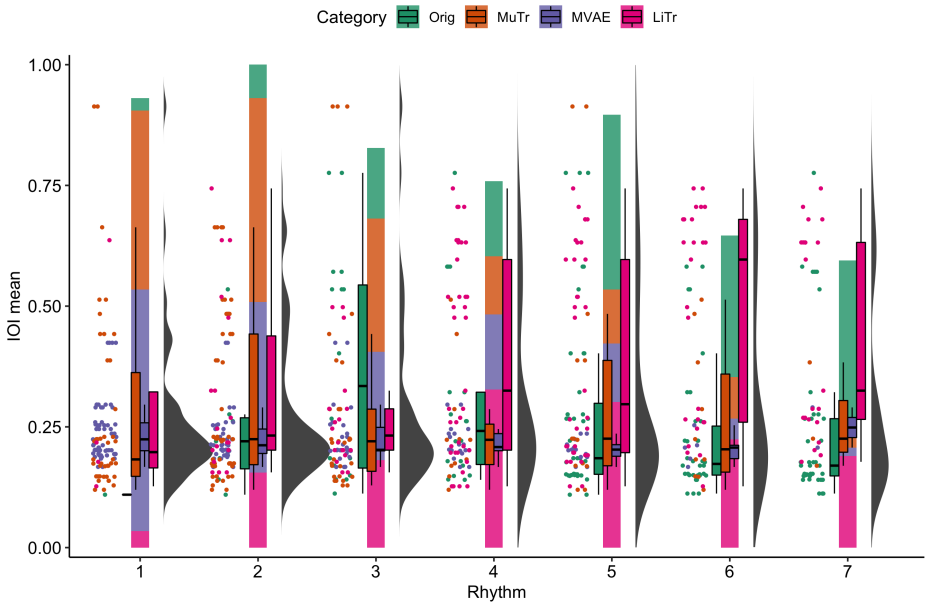


Figure B.102: IOI mean against Rhythm ratings.

B.2.5 IOI variance

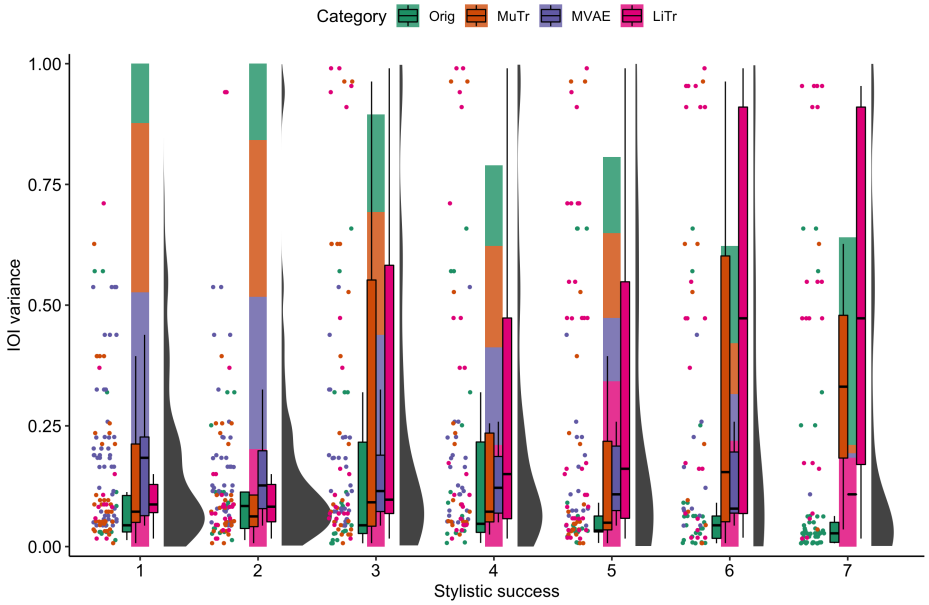


Figure B.103: IOI variance against Stylistic success ratings.

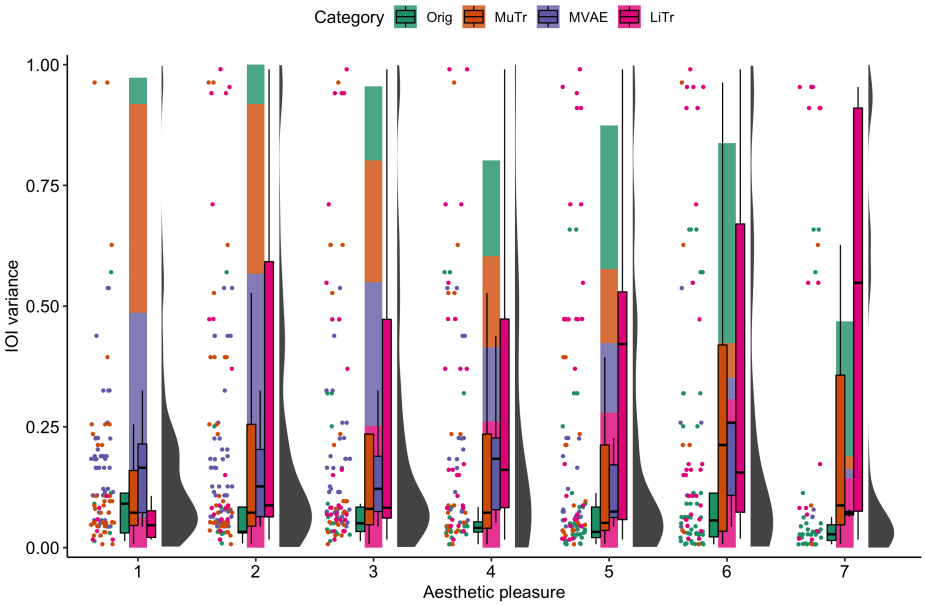


Figure B.104: IOI variance against Aesthetic pleasure ratings.

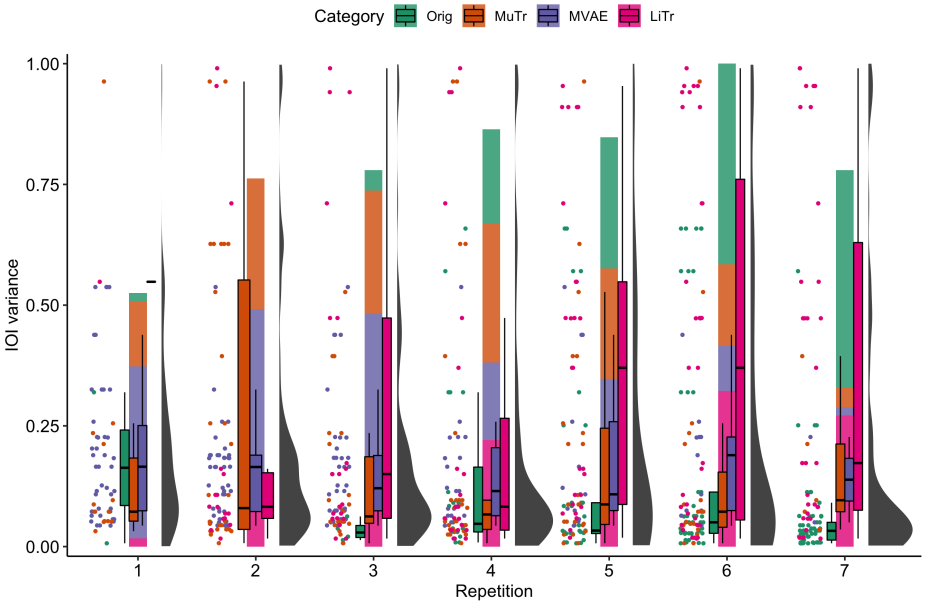


Figure B.105: IOI variance against Repetition ratings.

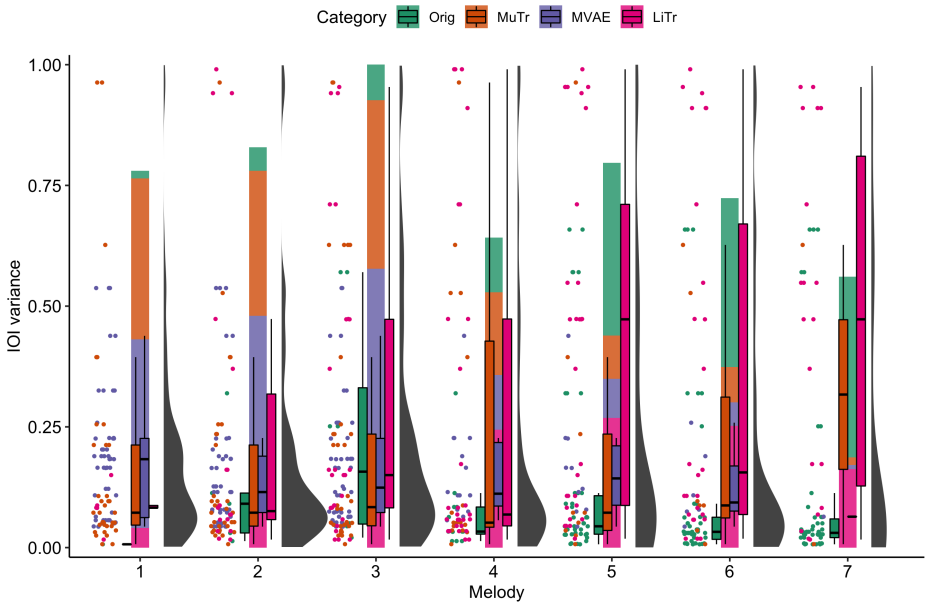


Figure B.106: IOI variance against Melody ratings.

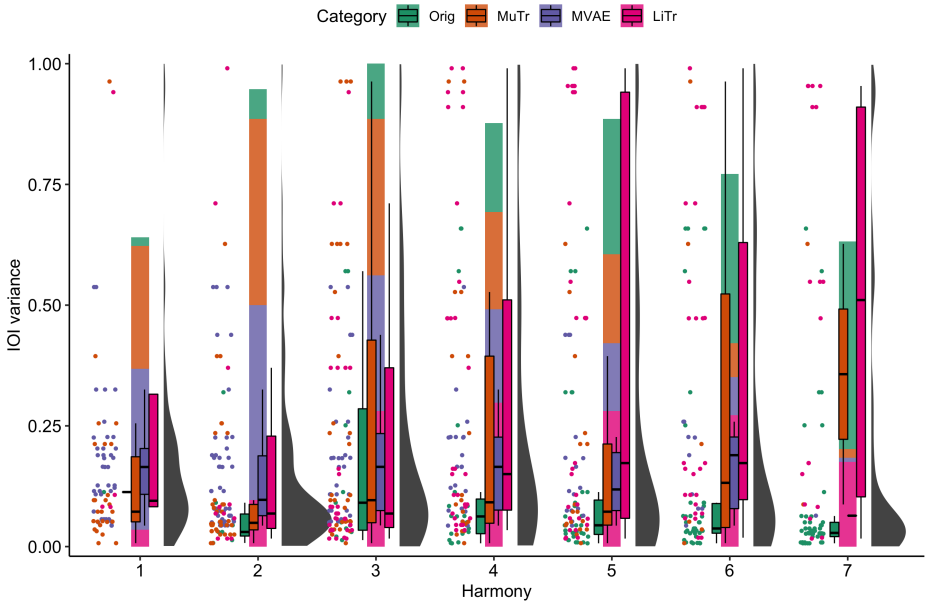


Figure B.107: IOI variance against Harmony ratings.

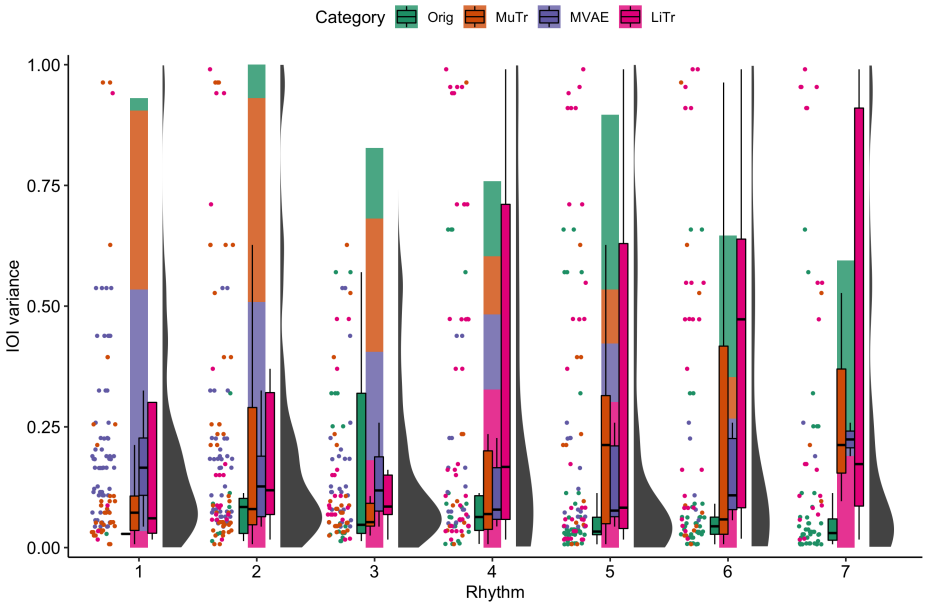


Figure B.108: IOI variance against Rhythm ratings.

B.2.6 IOI2 mean

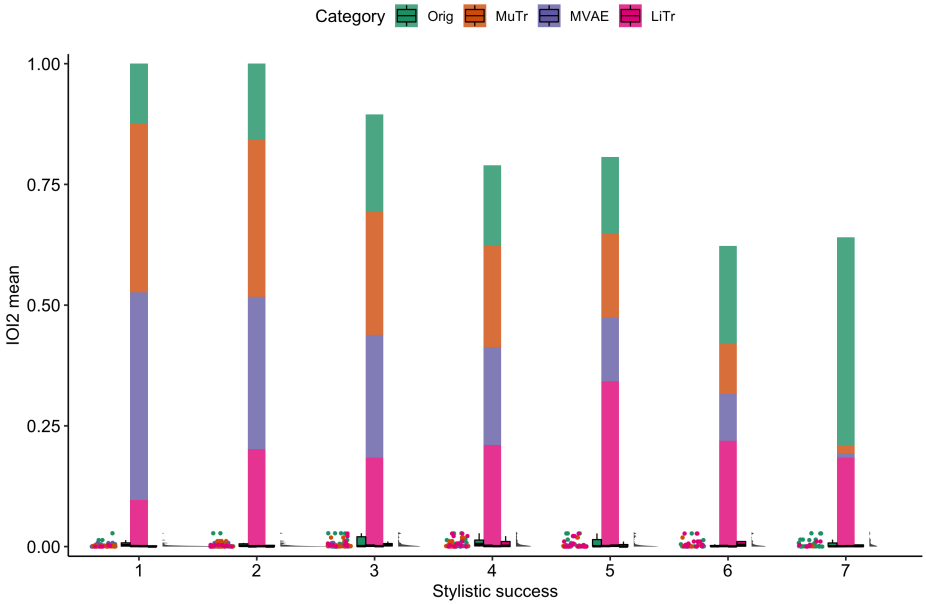


Figure B.109: IOI2 mean against Stylistic success ratings.

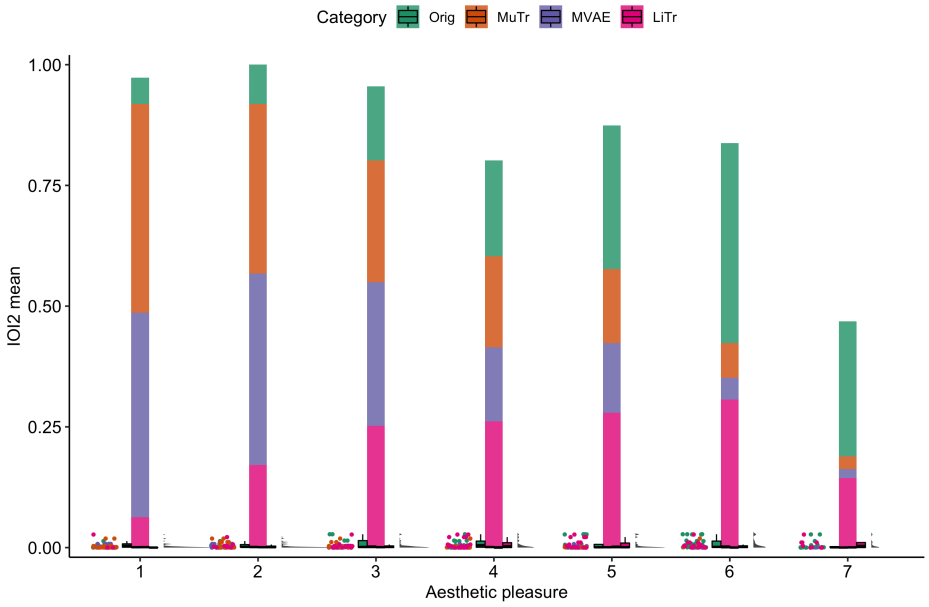


Figure B.110: IOI2 mean against Aesthetic pleasure ratings.

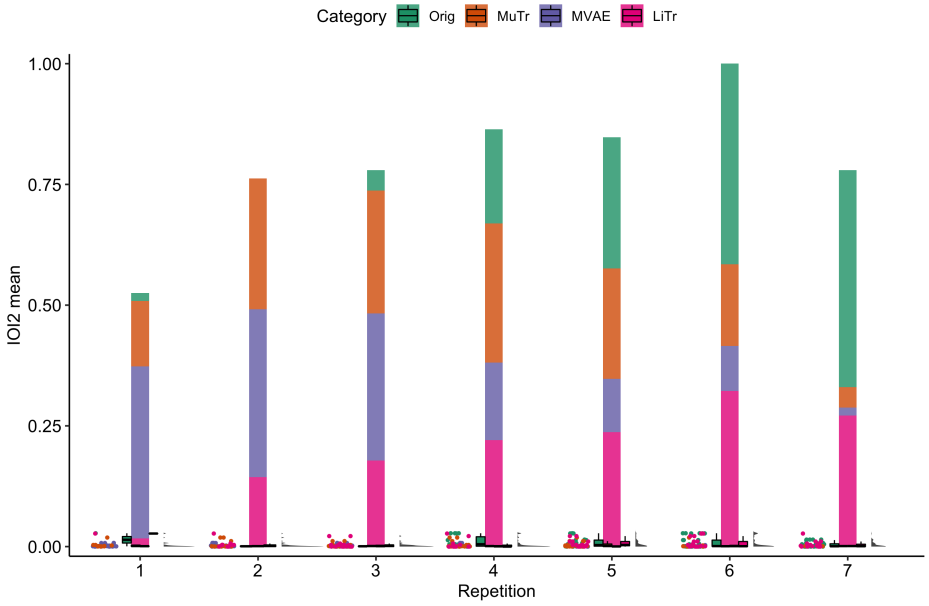


Figure B.111: IOI2 mean against Repetition ratings.

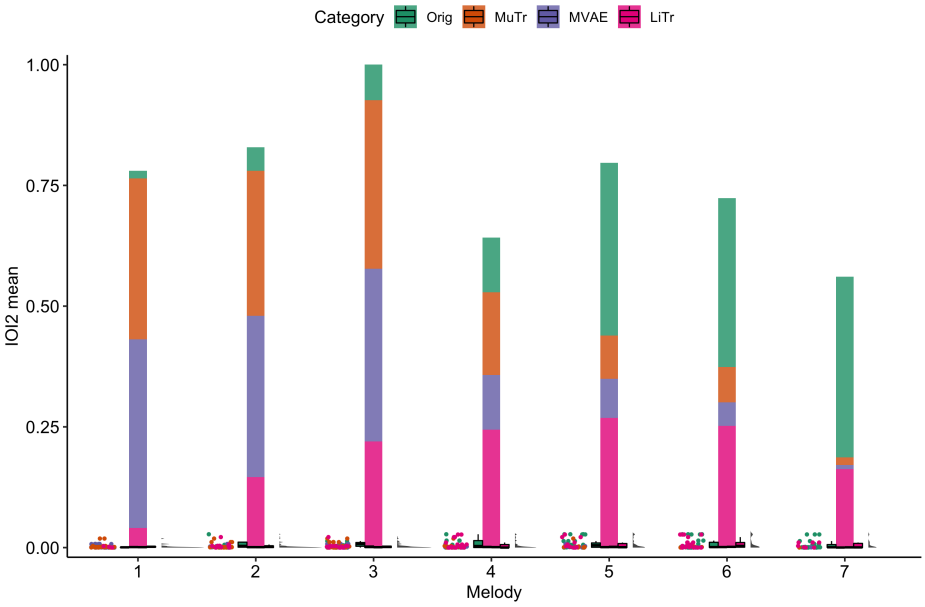


Figure B.112: IOI2 mean against Melody ratings.

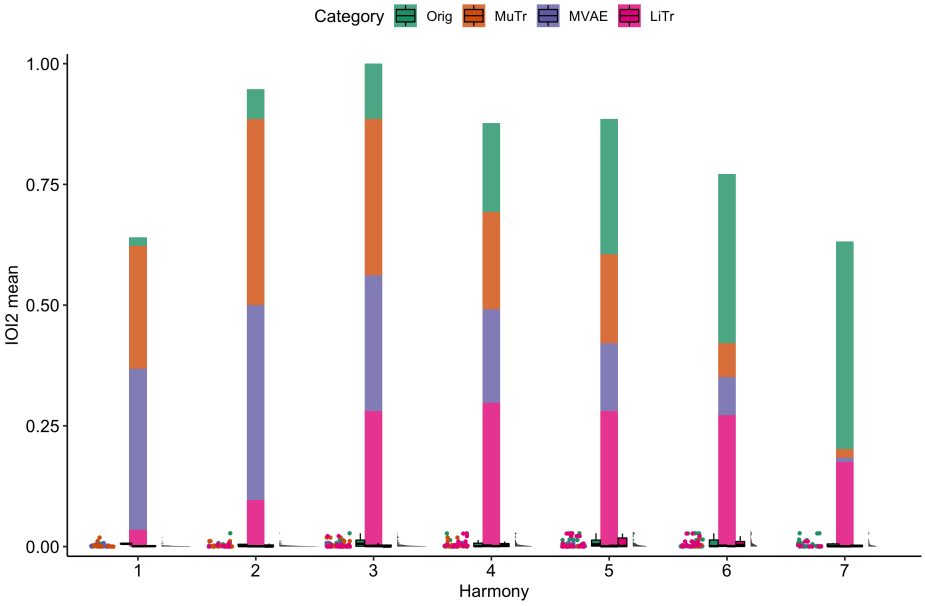


Figure B.113: IOI2 mean against Harmony ratings.

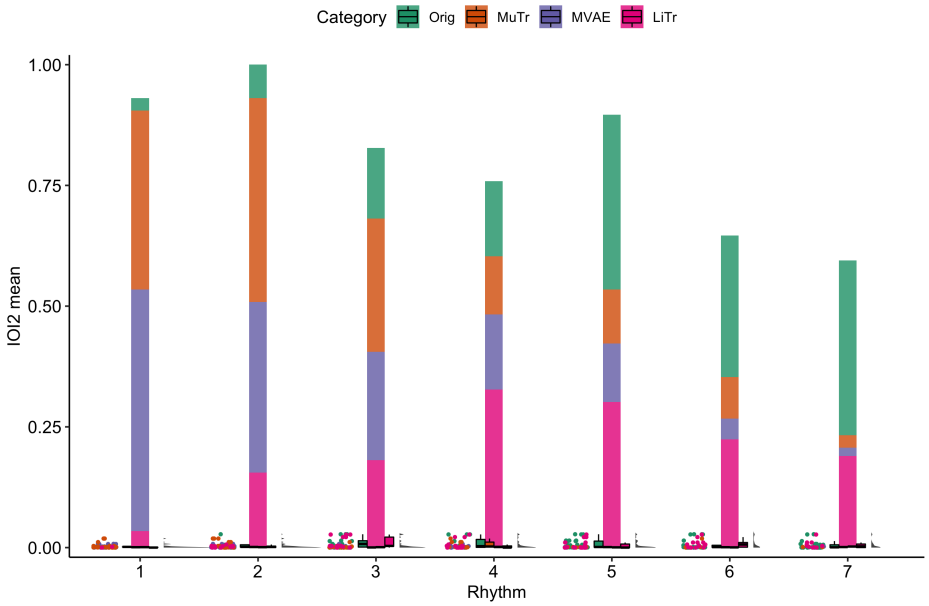


Figure B.114: IOI2 mean against Rhythm ratings.

B.2.7 IOI2 variance

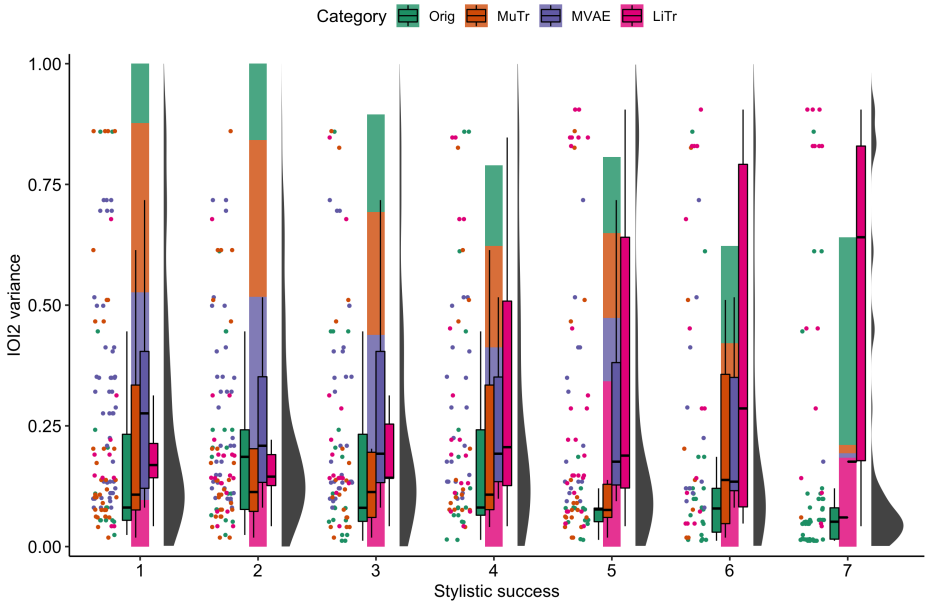


Figure B.115: IOI2 variance against Stylistic success ratings.

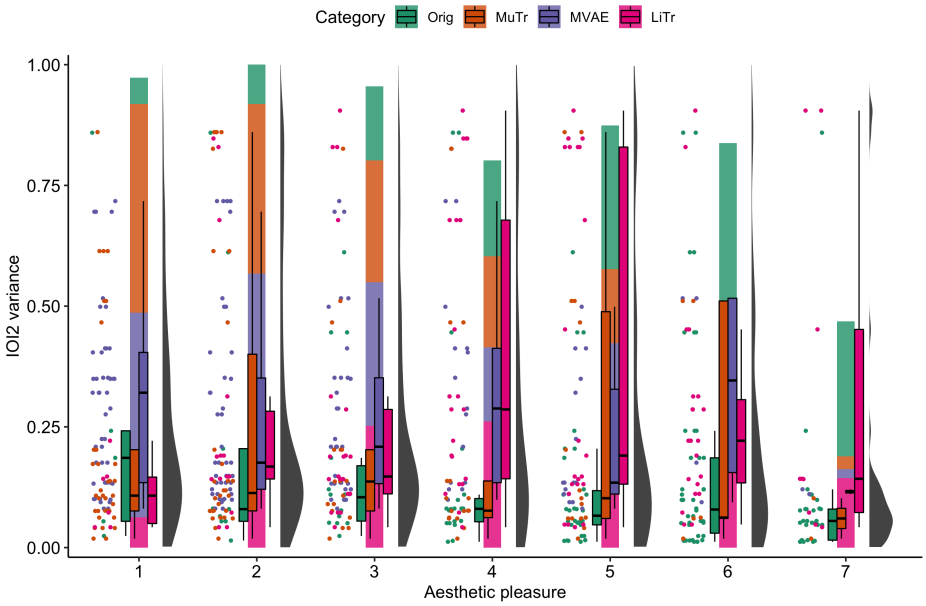


Figure B.116: IOI2 variance against Aesthetic pleasure ratings.

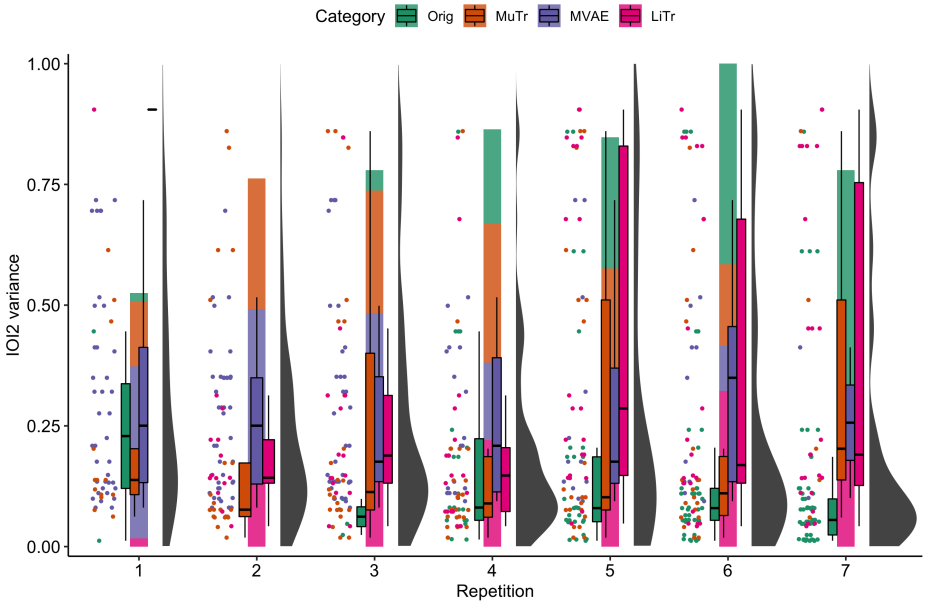


Figure B.117: IOI2 variance against Repetition ratings.

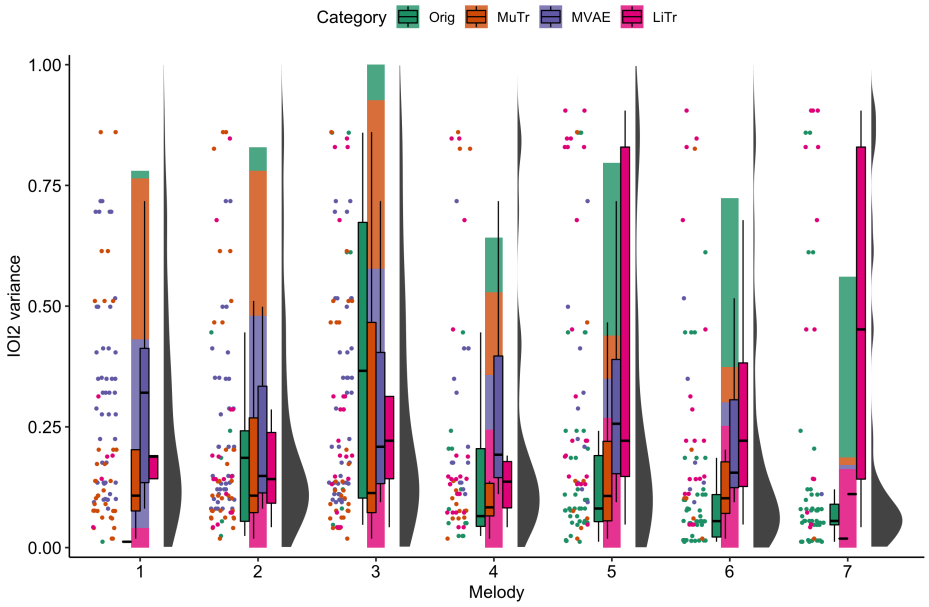


Figure B.118: IOI2 variance against Melody ratings.

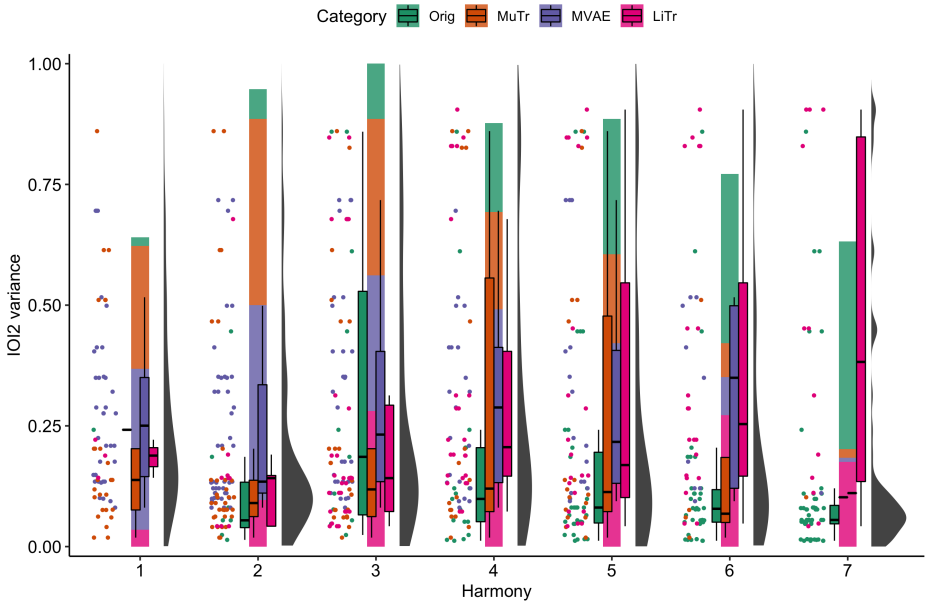


Figure B.119: IOI2 variance against Harmony ratings.

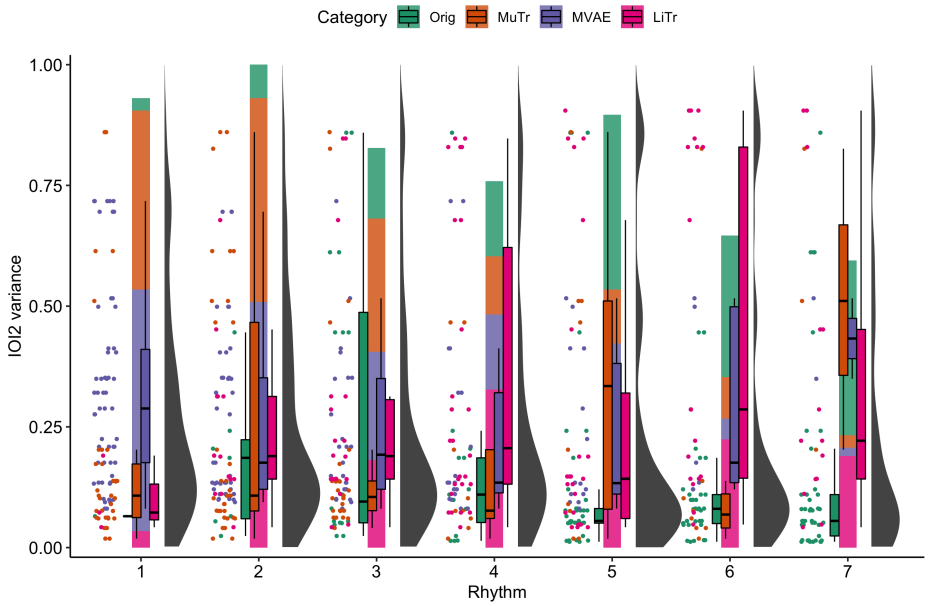


Figure B.120: IOI2 variance against Rhythm ratings.

B.2.8 KOT mean

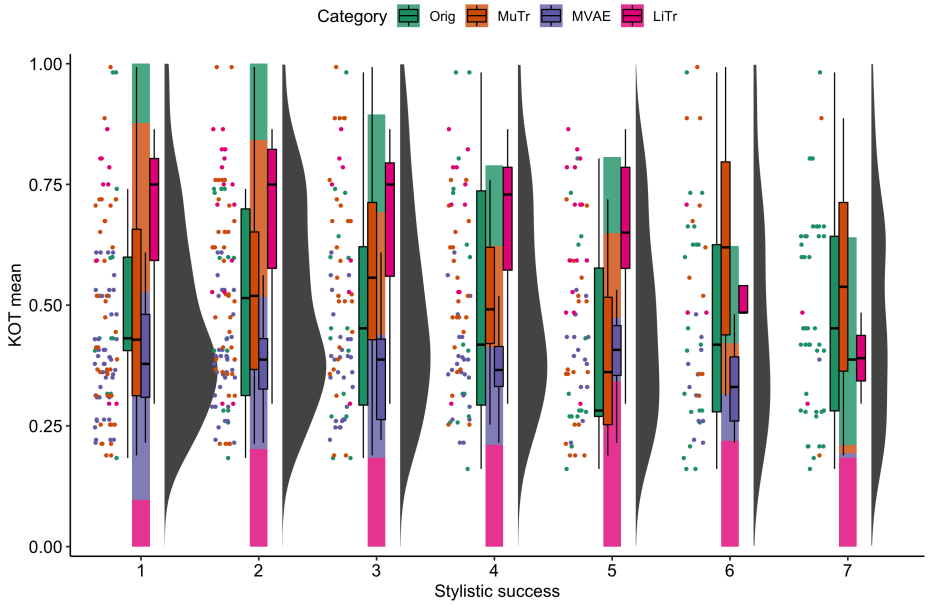


Figure B.121: KOT mean against Stylistic success ratings.

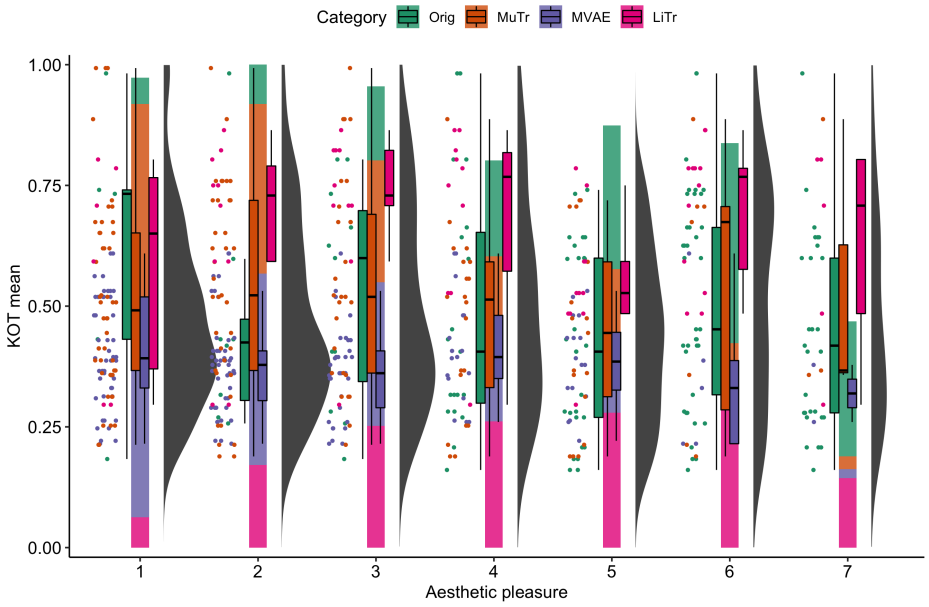


Figure B.122: KOT mean against Aesthetic pleasure ratings.

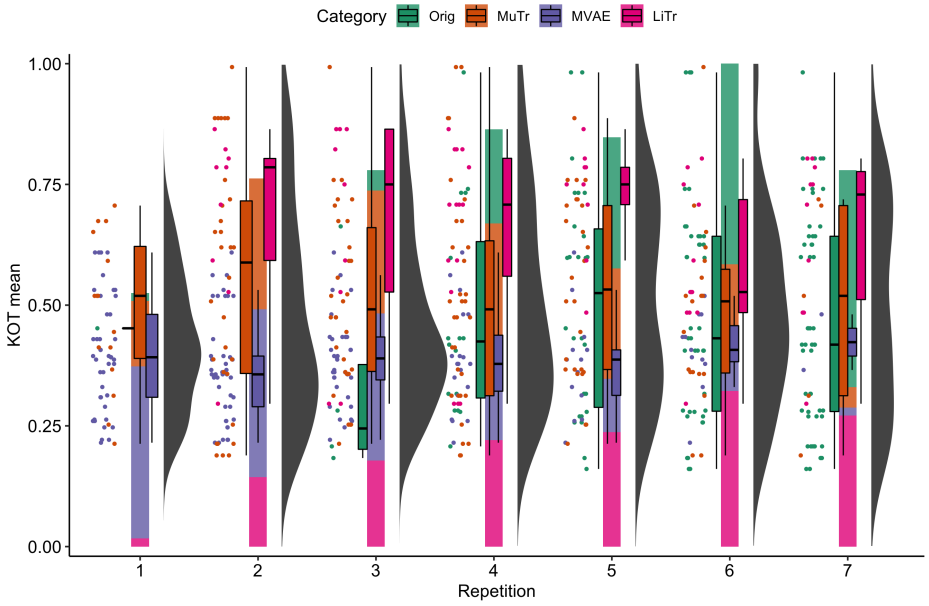


Figure B.123: KOT mean against Repetition ratings.

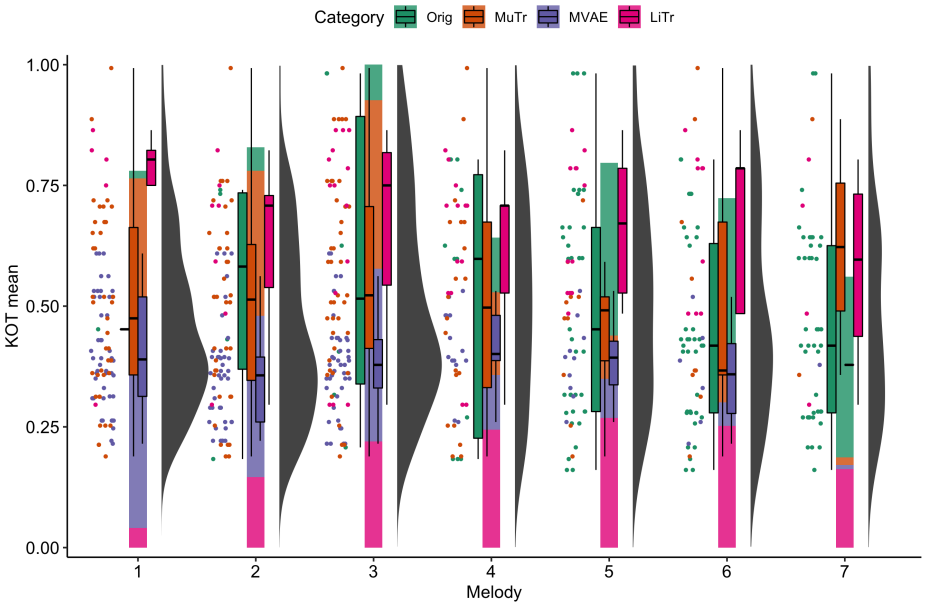


Figure B.124: KOT mean against Melody ratings.

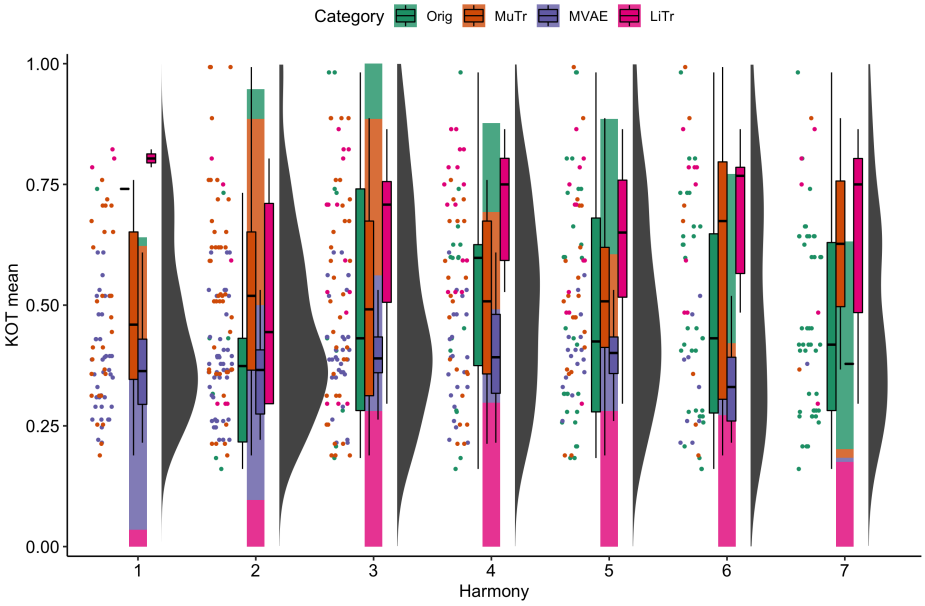


Figure B.125: KOT mean against Harmony ratings.

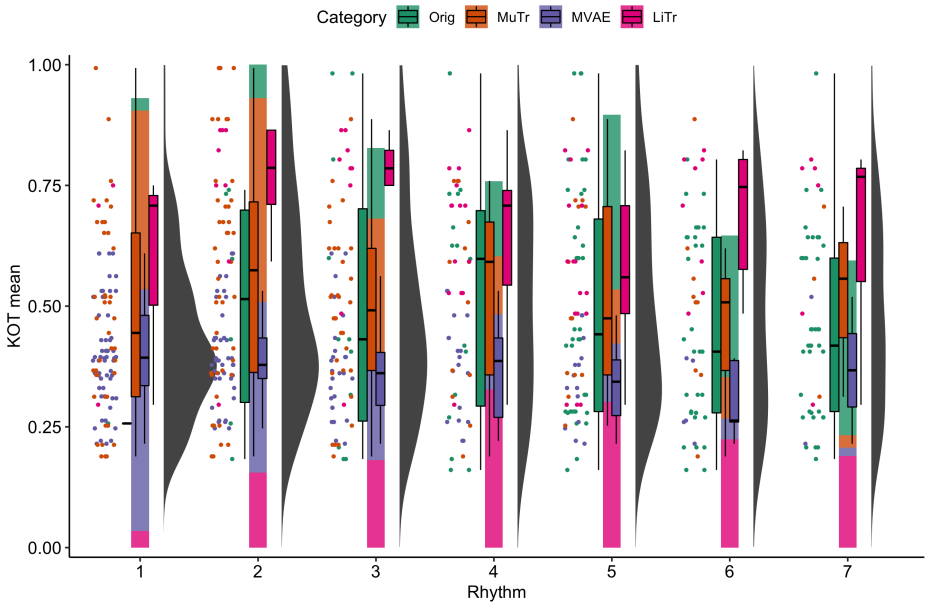


Figure B.126: KOT mean against Rhythm ratings.

B.2.9 KOT variance

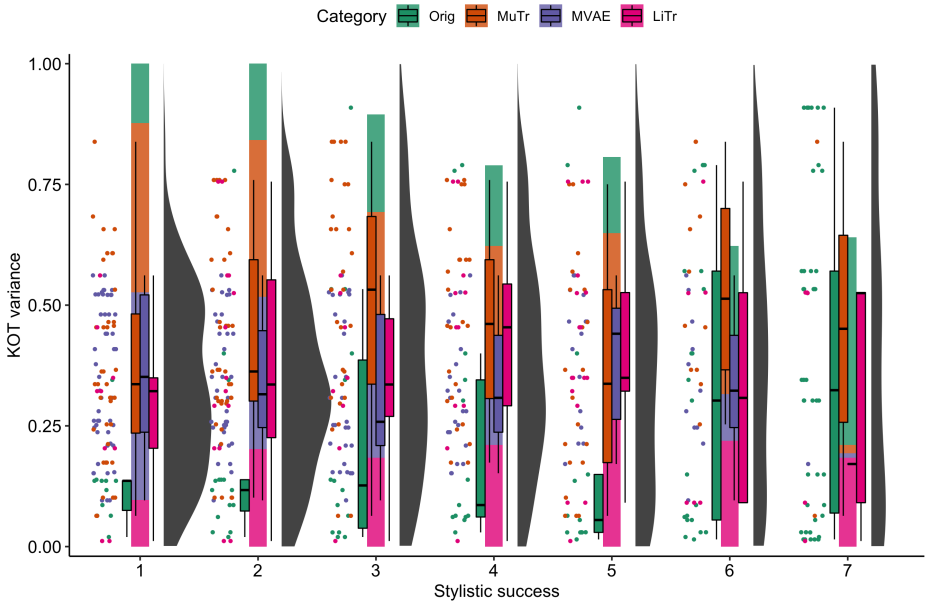


Figure B.127: KOT variance against Stylistic success ratings.

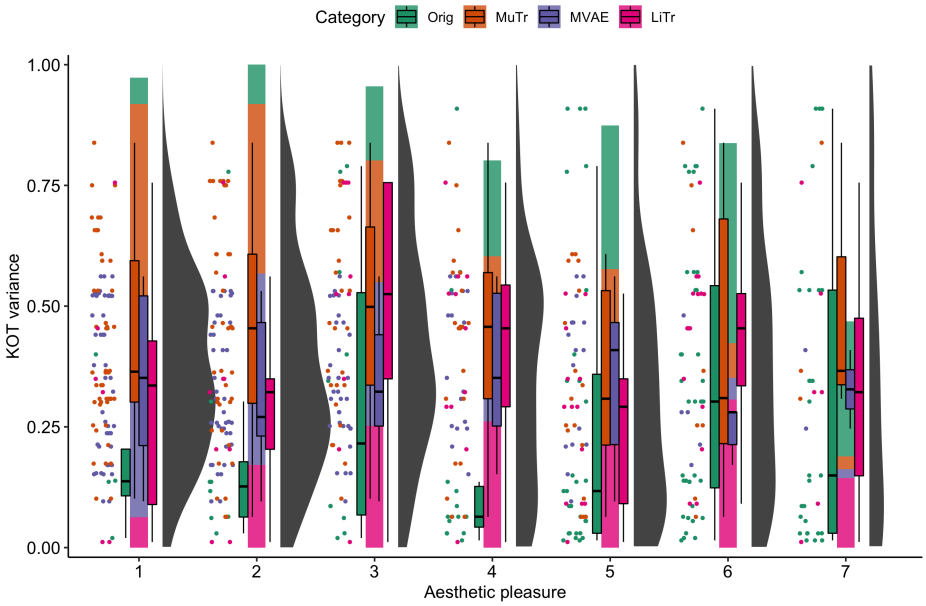


Figure B.128: KOT variance against Aesthetic pleasure ratings.

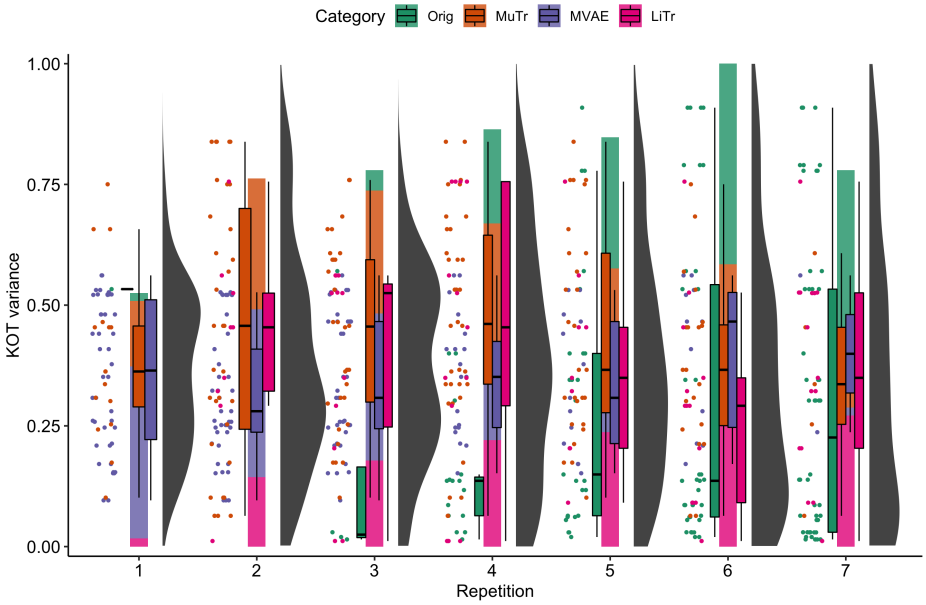


Figure B.129: KOT variance against Repetition ratings.

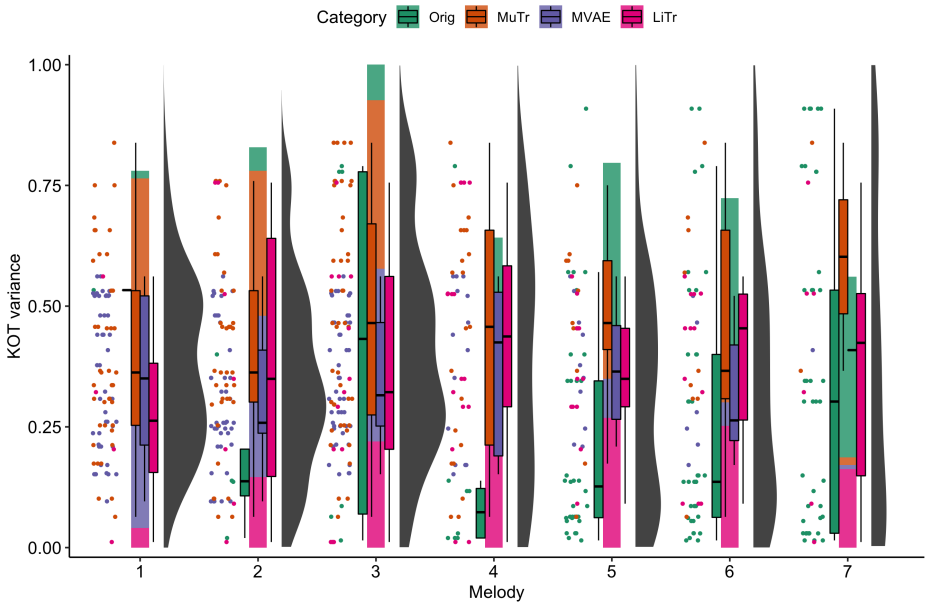


Figure B.130: KOT variance against Melody ratings.

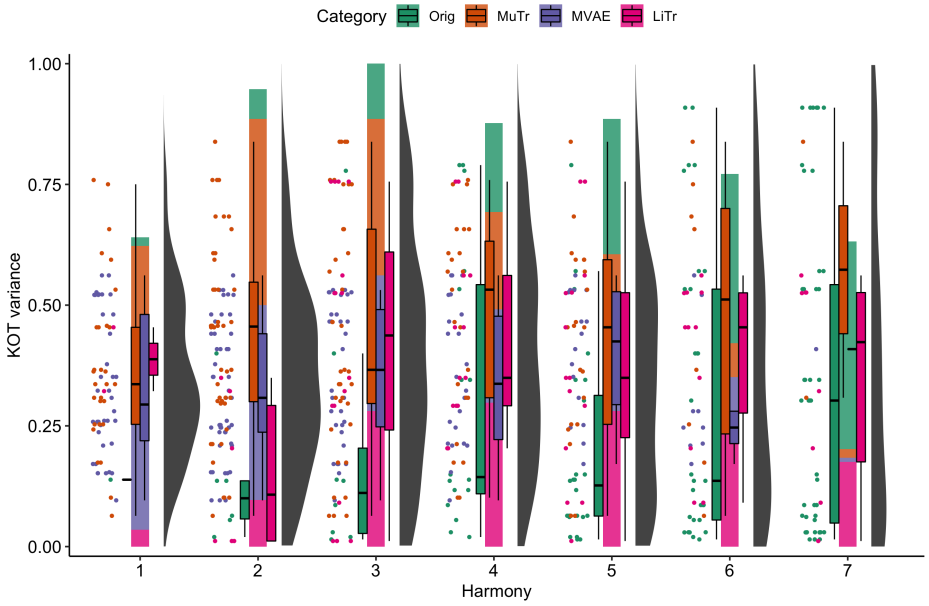


Figure B.131: KOT variance against Harmony ratings.

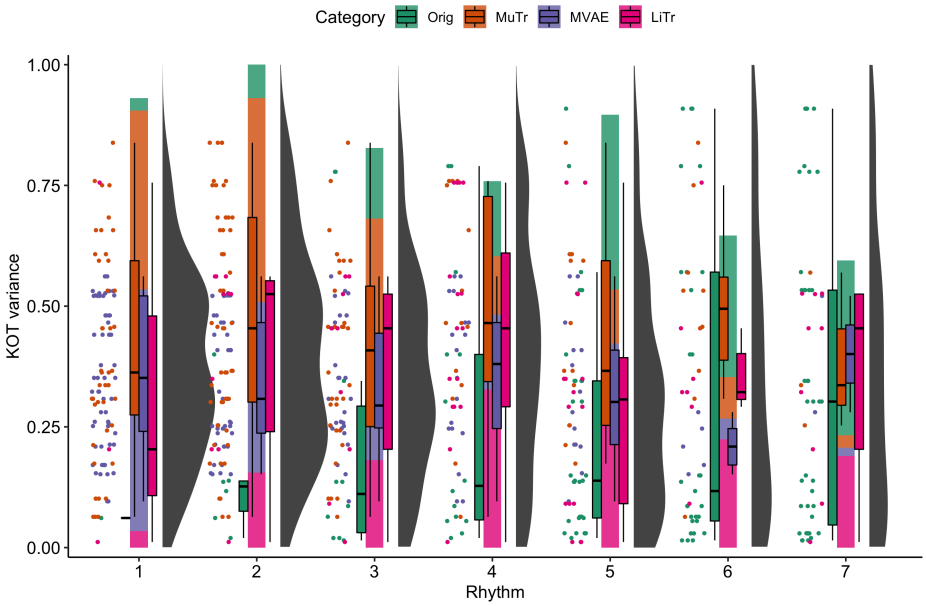


Figure B.132: KOT variance against Rhythm ratings.

B.2.10 KDT mean

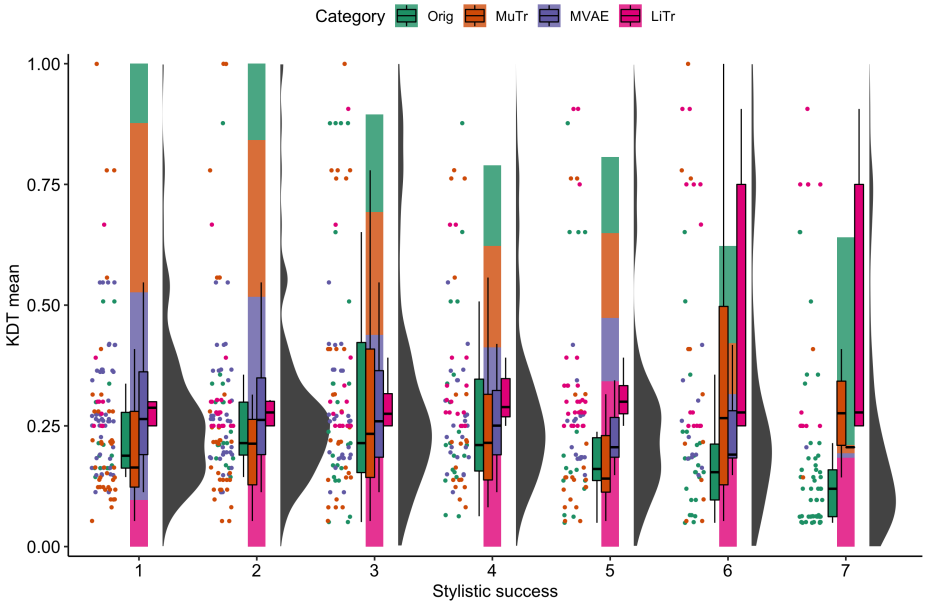


Figure B.133: KDT mean against Stylistic success ratings.

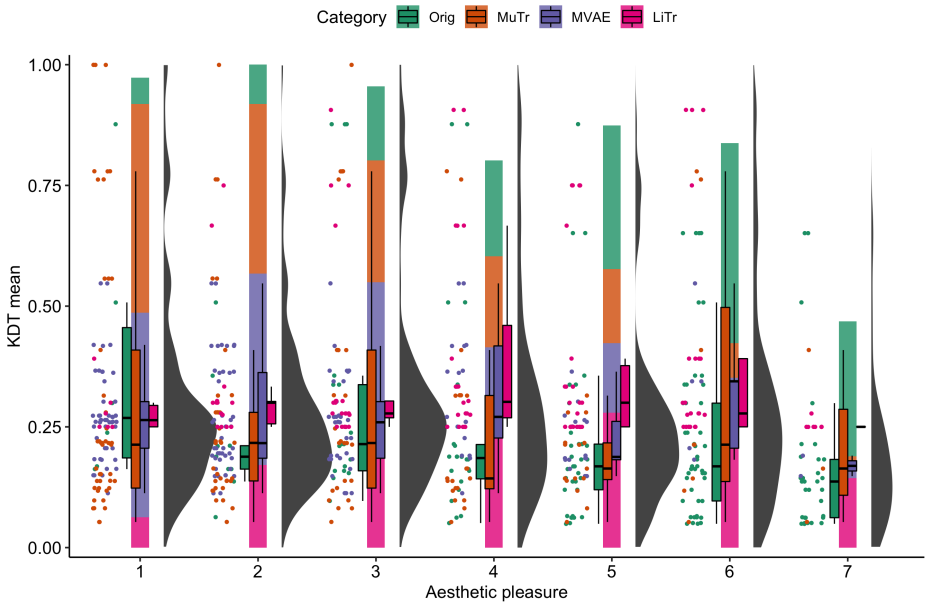


Figure B.134: KDT mean against Aesthetic pleasure ratings.

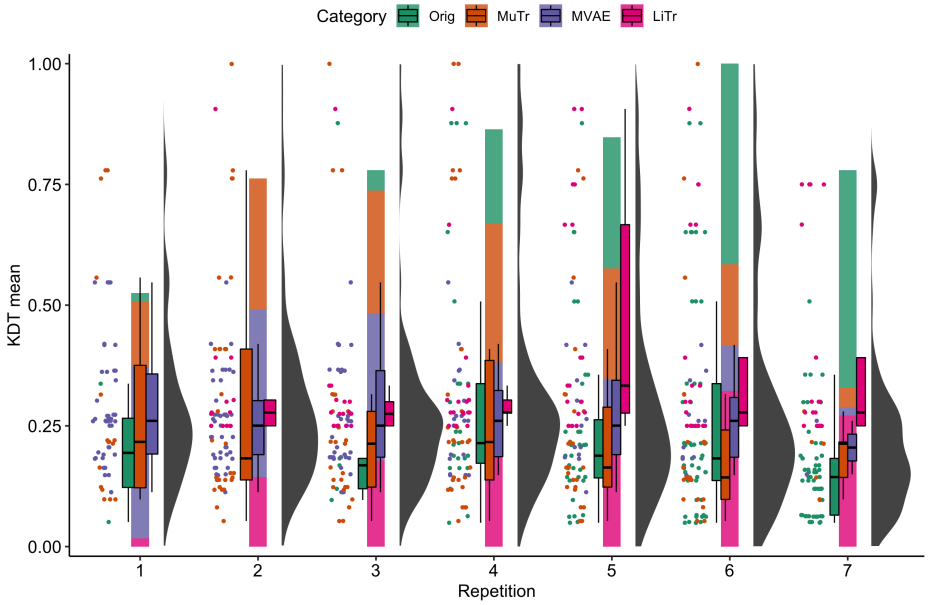


Figure B.135: KDT mean against Repetition ratings.

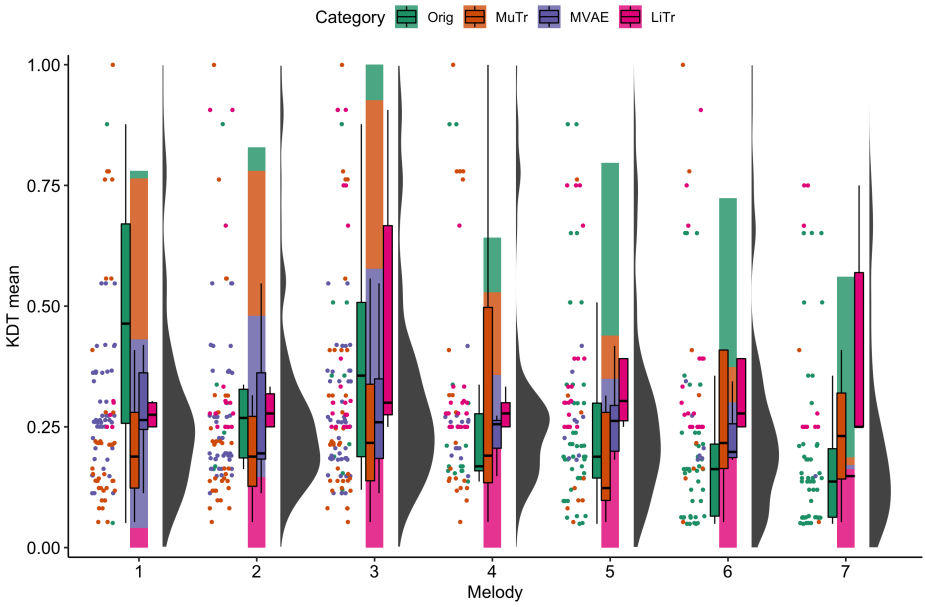


Figure B.136: KDT mean against Melody ratings.

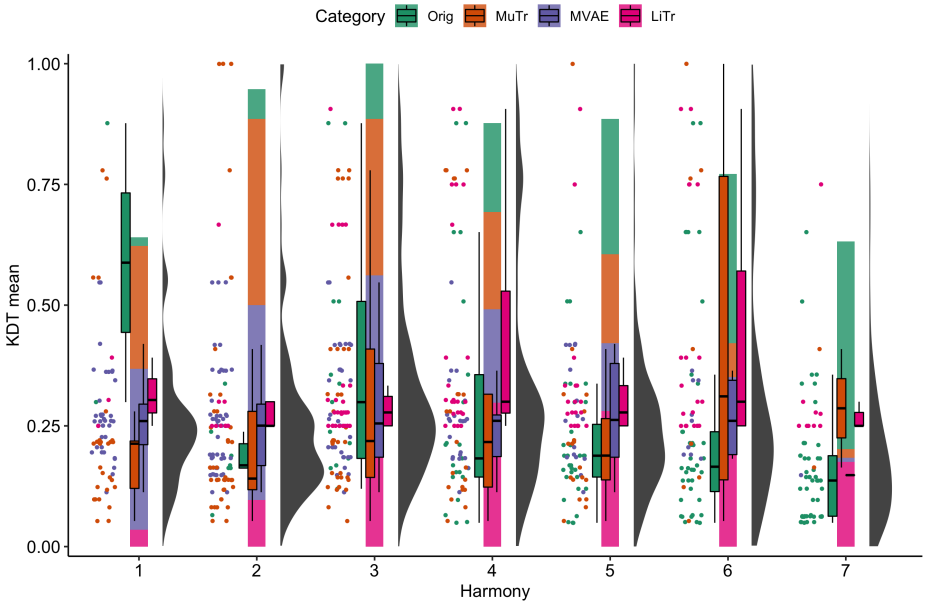


Figure B.137: KDT mean against Harmony ratings.

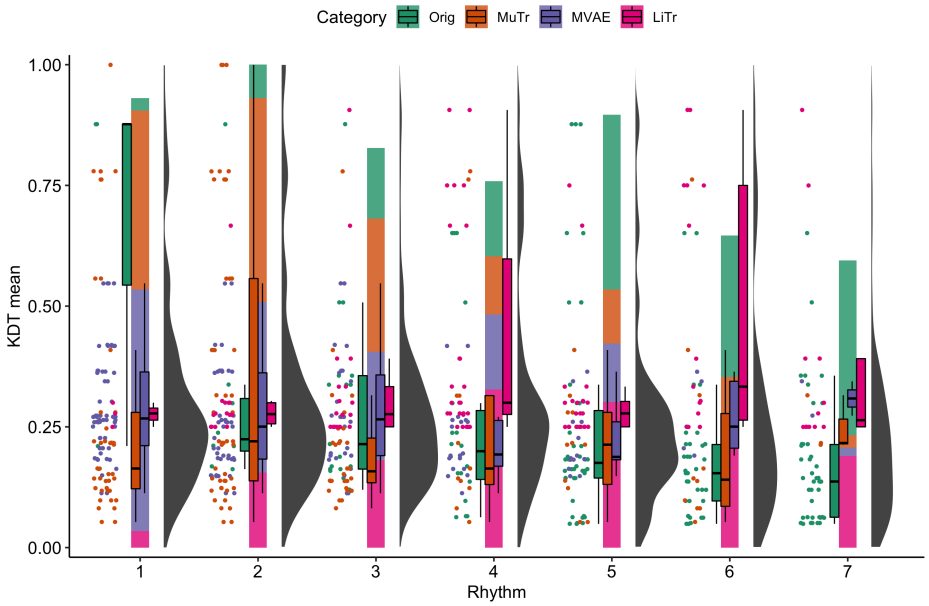


Figure B.138: KDT mean against Rhythm ratings.

B.2.11 KDT variance

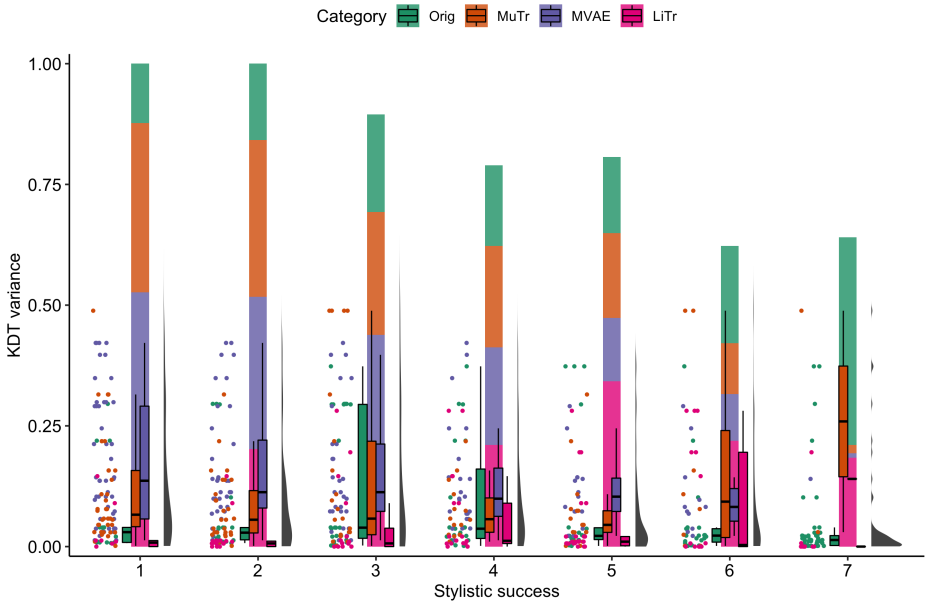


Figure B.139: KDT variance against Stylistic success ratings.

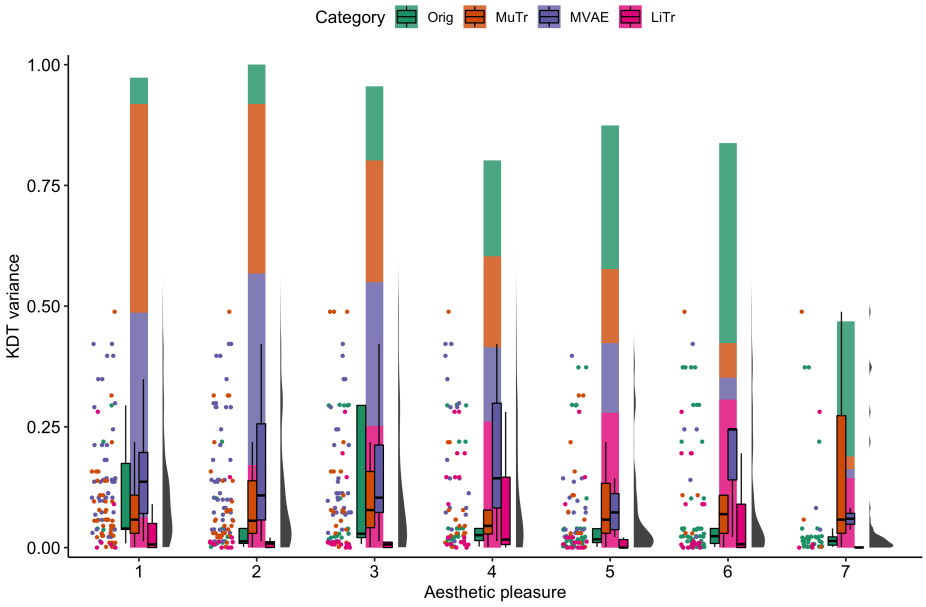


Figure B.140: KDT variance against Aesthetic pleasure ratings.

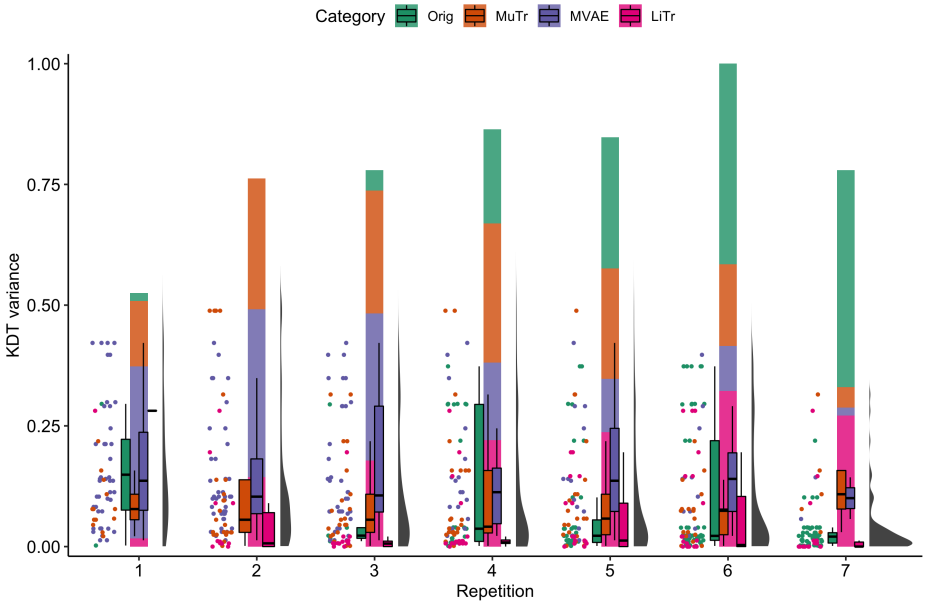


Figure B.141: KDT variance against Repetition ratings.

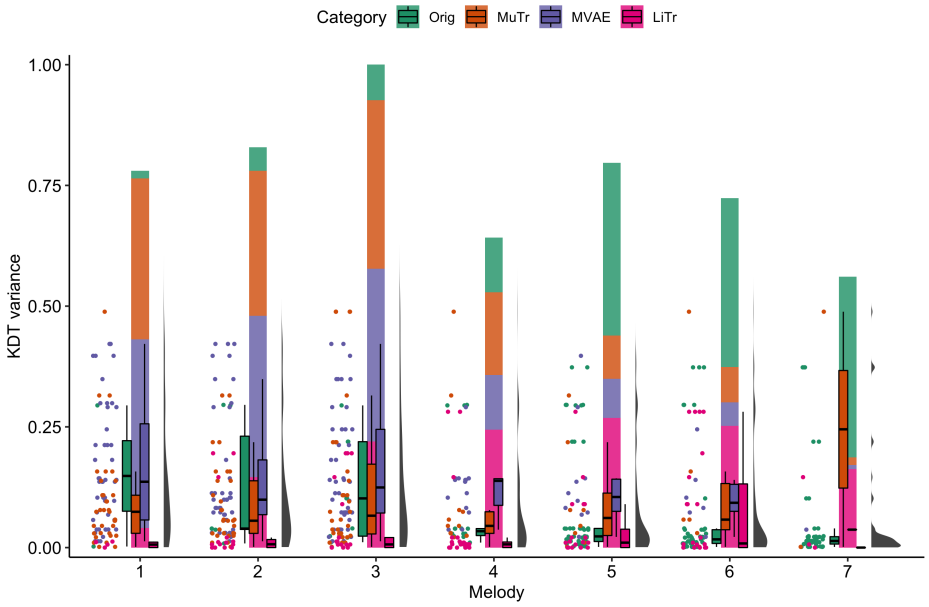


Figure B.142: KDT variance against Melody ratings.

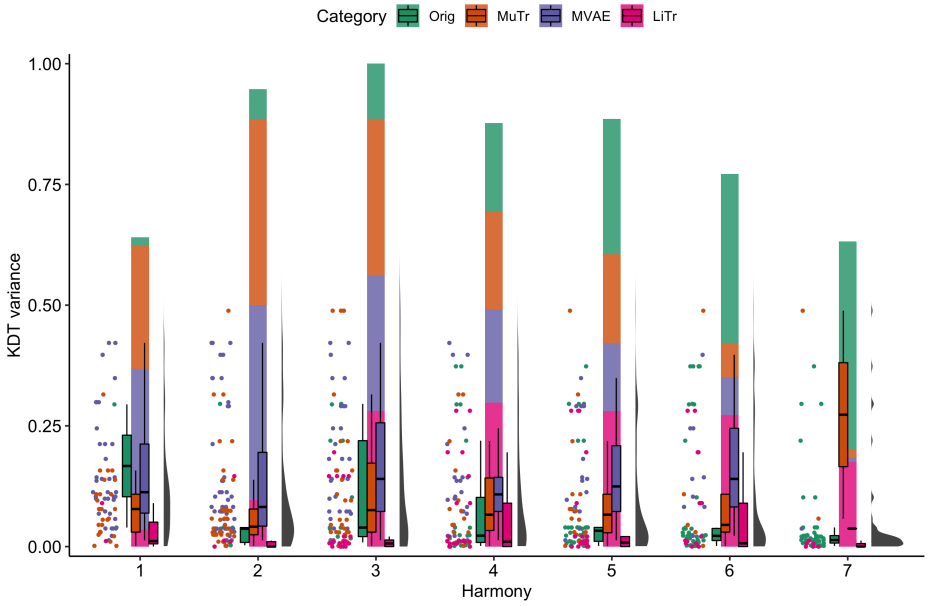


Figure B.143: KDT variance against Harmony ratings.

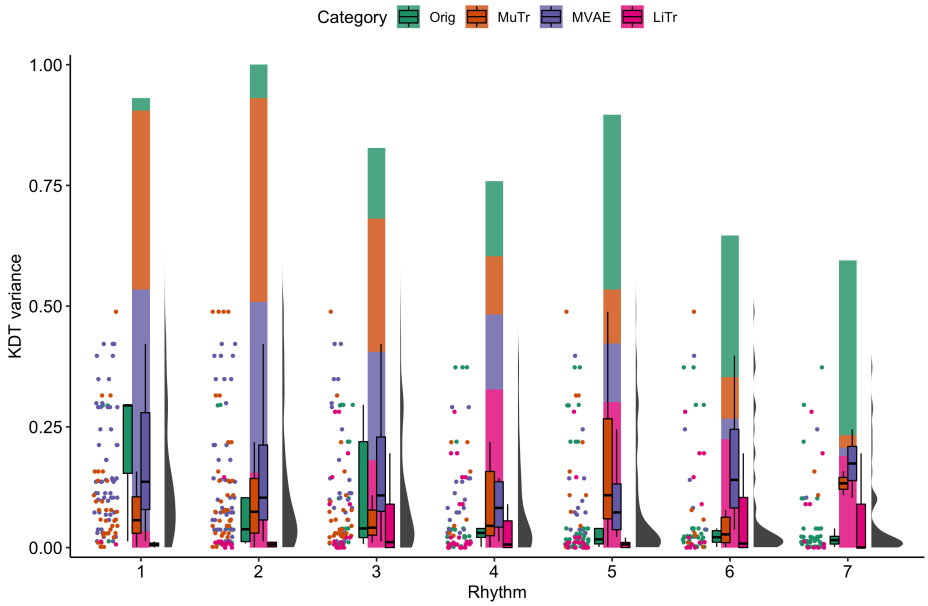


Figure B.144: KDT variance against Rhythm ratings.

B.2.12 Jitter mean

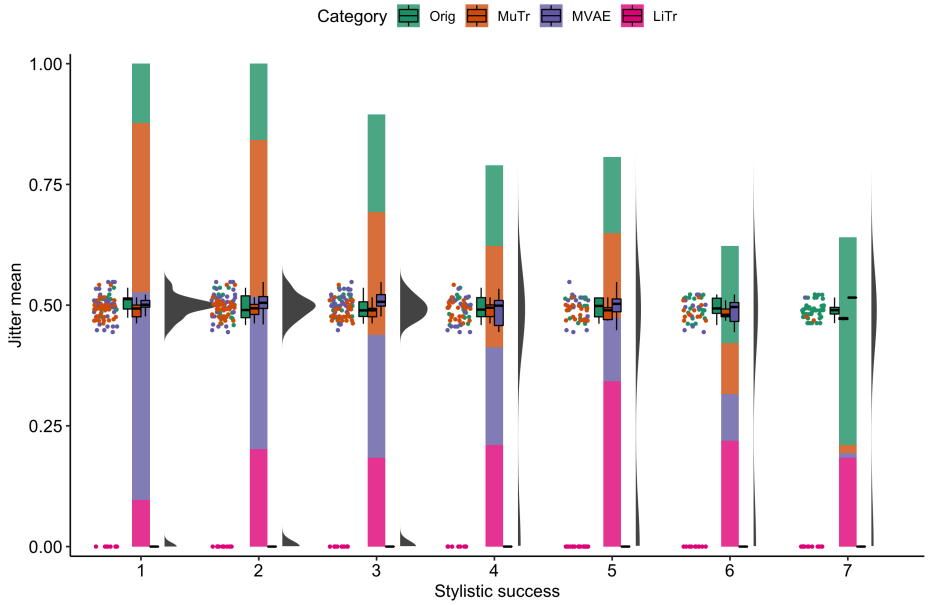


Figure B.145: Jitter mean against Stylistic success ratings.

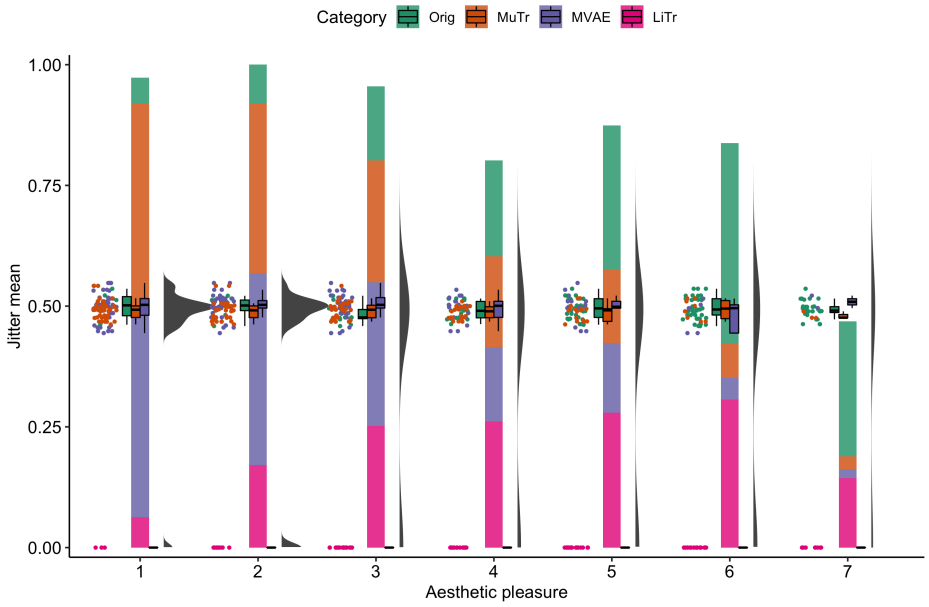


Figure B.146: Jitter mean against Aesthetic pleasure ratings.

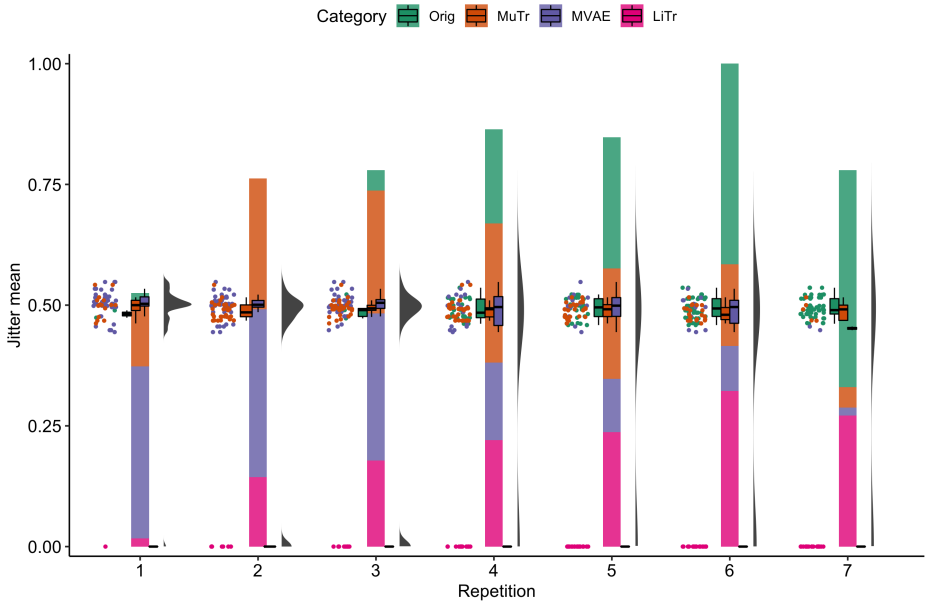


Figure B.147: Jitter mean against Repetition ratings.

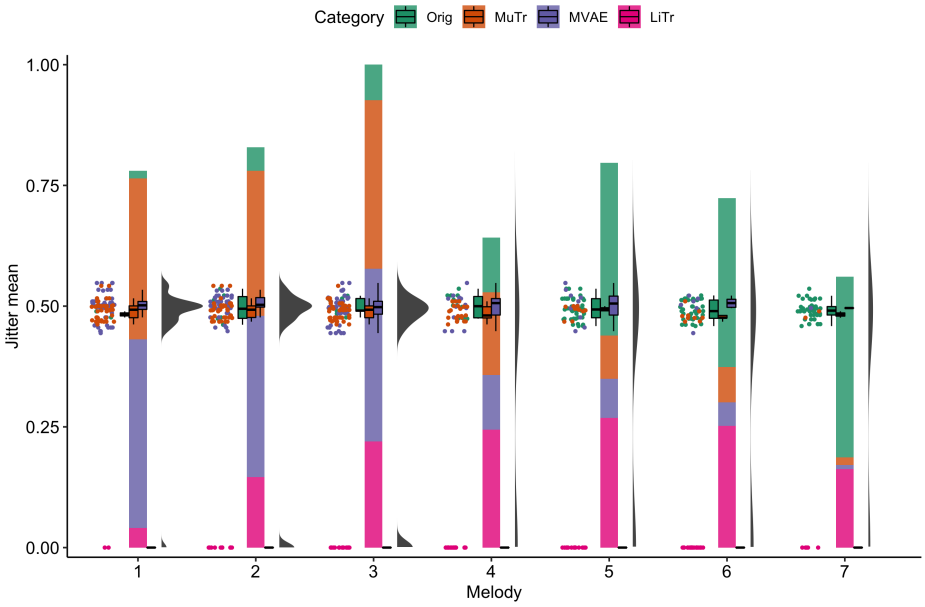


Figure B.148: Jitter mean against Melody ratings.

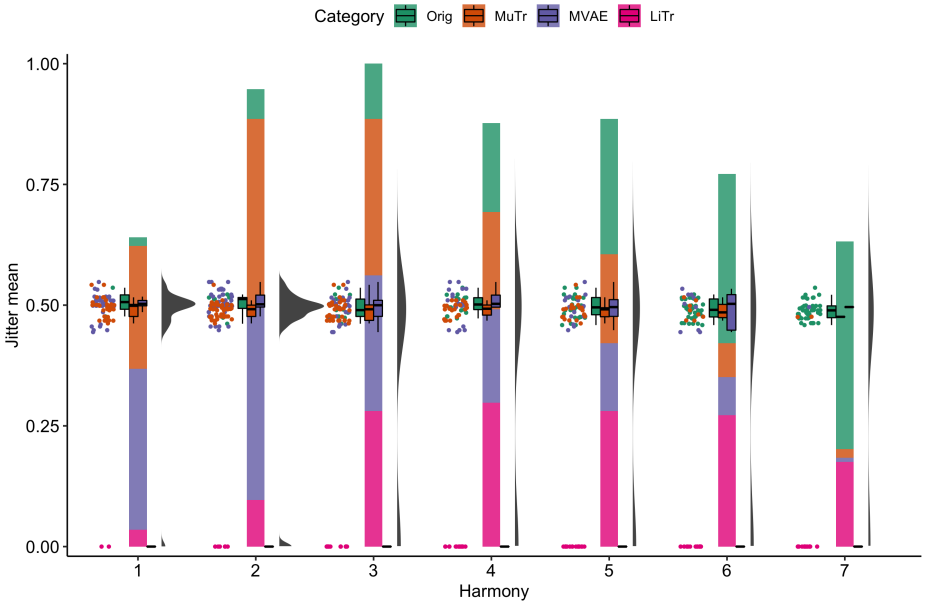


Figure B.149: Jitter mean against Harmony ratings.

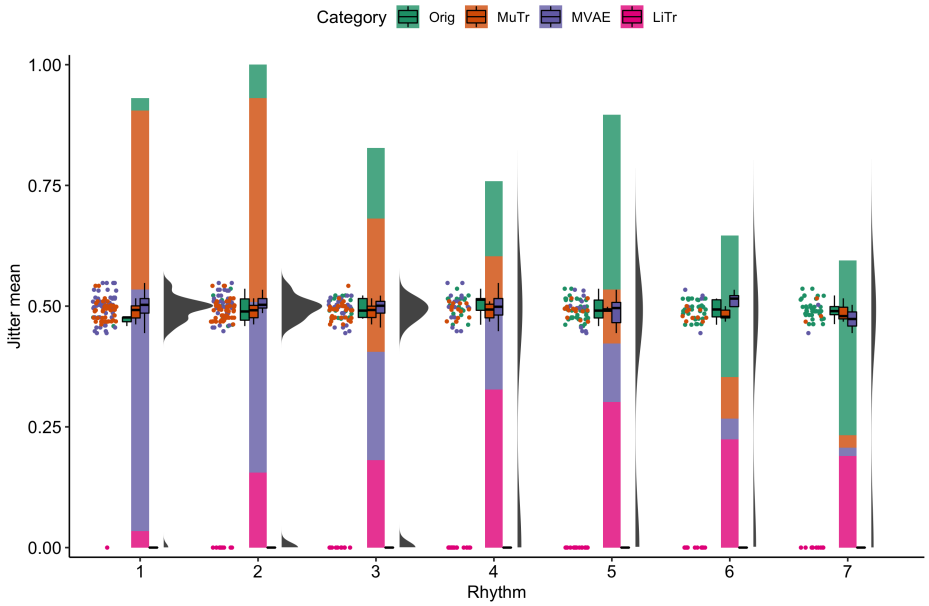


Figure B.150: Jitter mean against Rhythm ratings.

B.2.13 Jitter variance

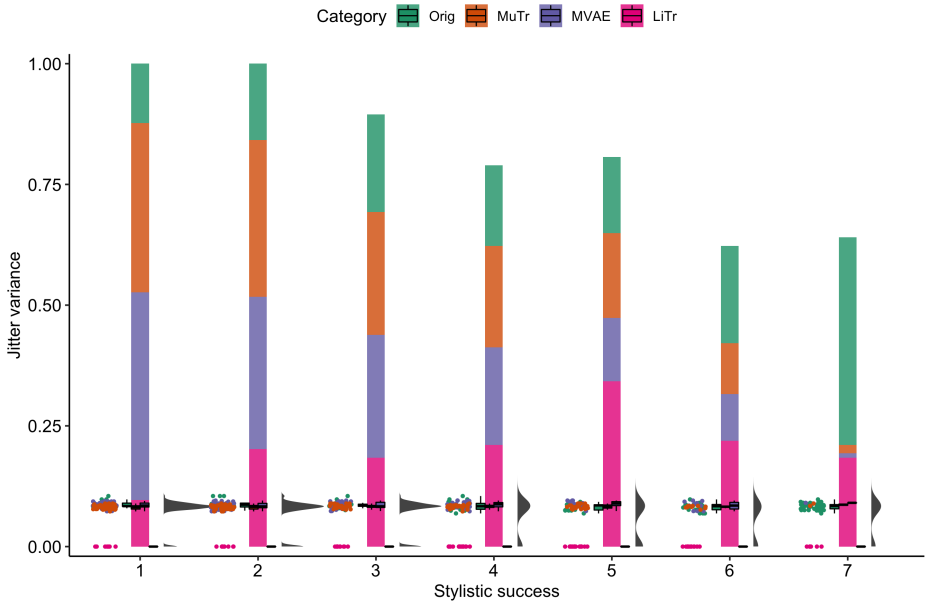


Figure B.151: Jitter variance against Stylistic success ratings.

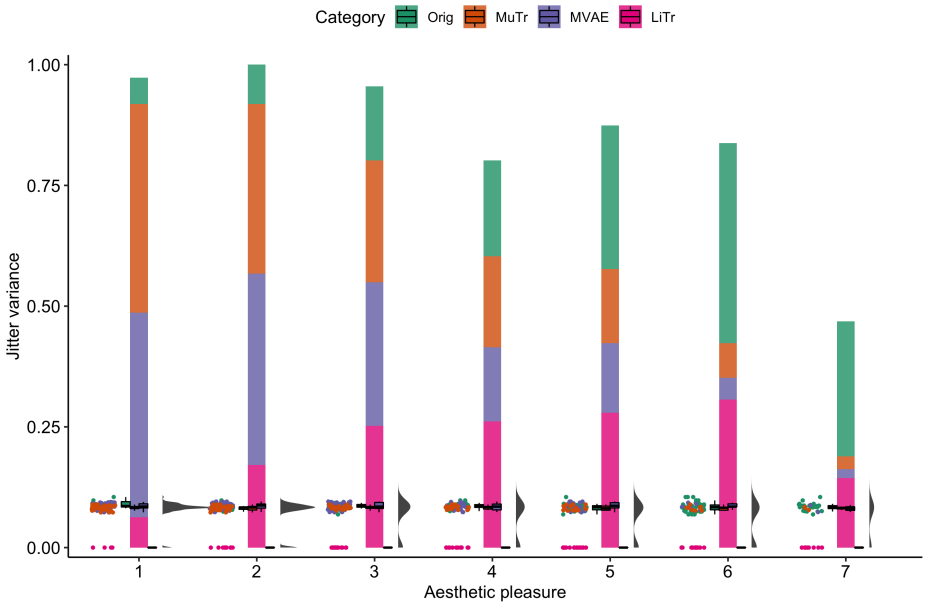


Figure B.152: Jitter variance against Aesthetic pleasure ratings.

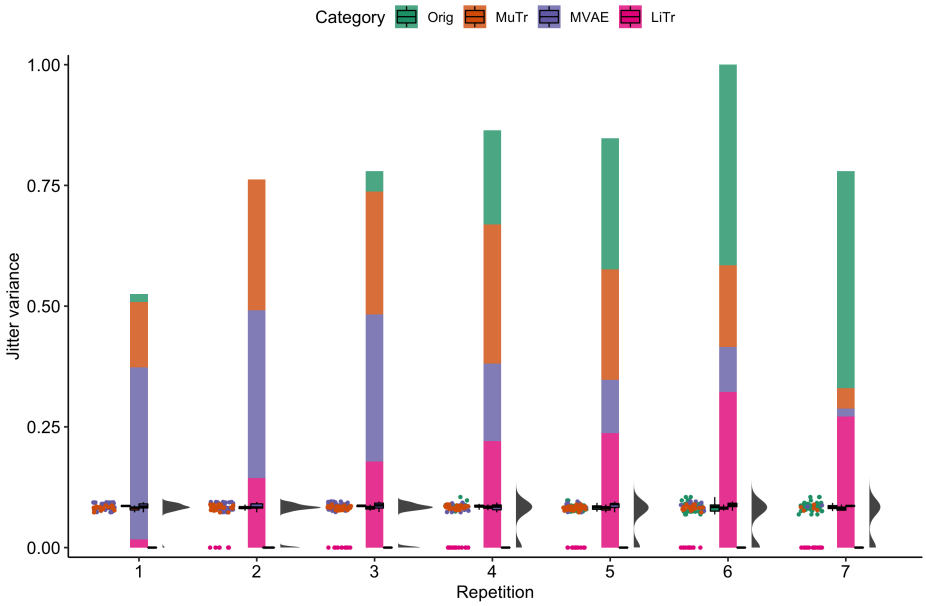


Figure B.153: Jitter variance against Repetition ratings.

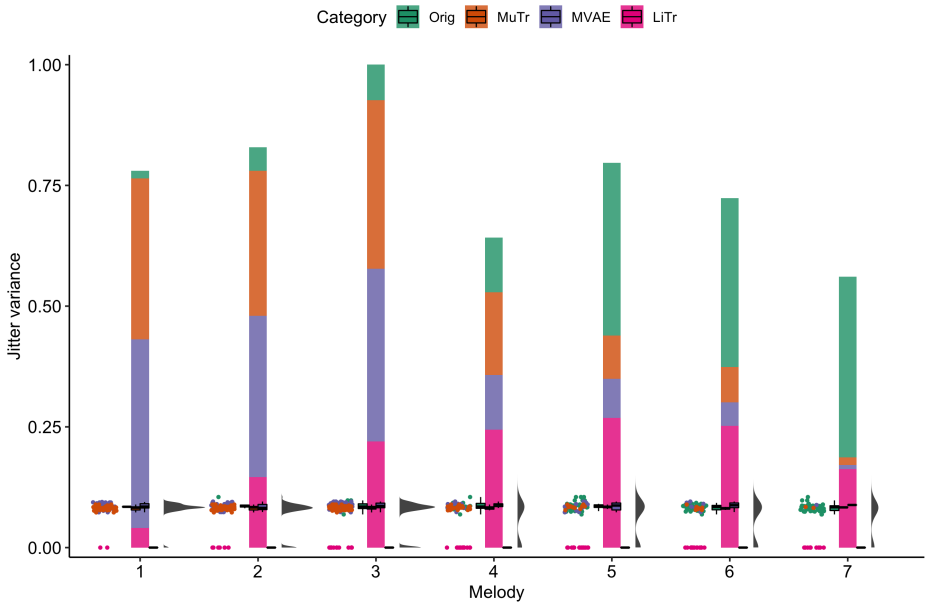


Figure B.154: Jitter variance against Melody ratings.

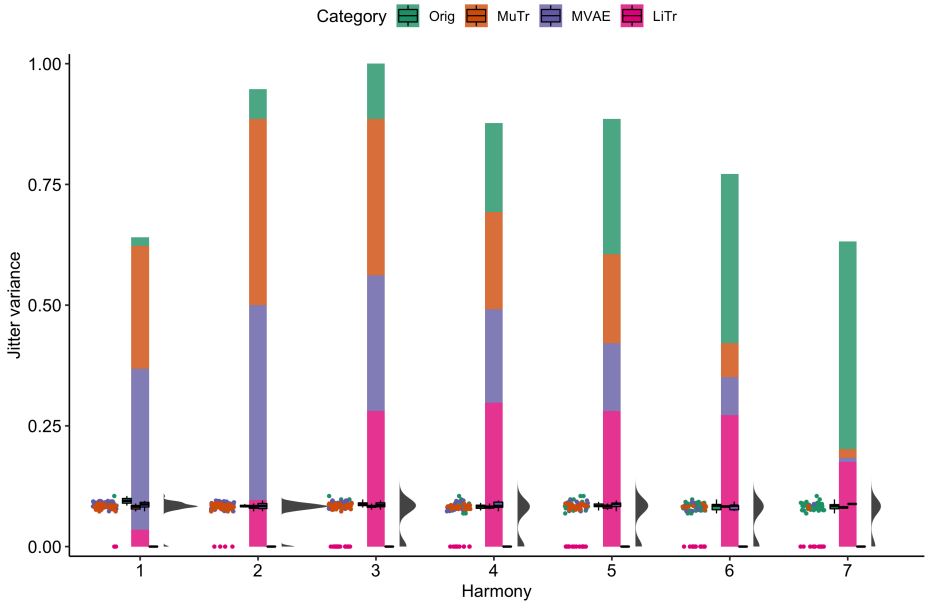


Figure B.155: Jitter variance against Harmony ratings.

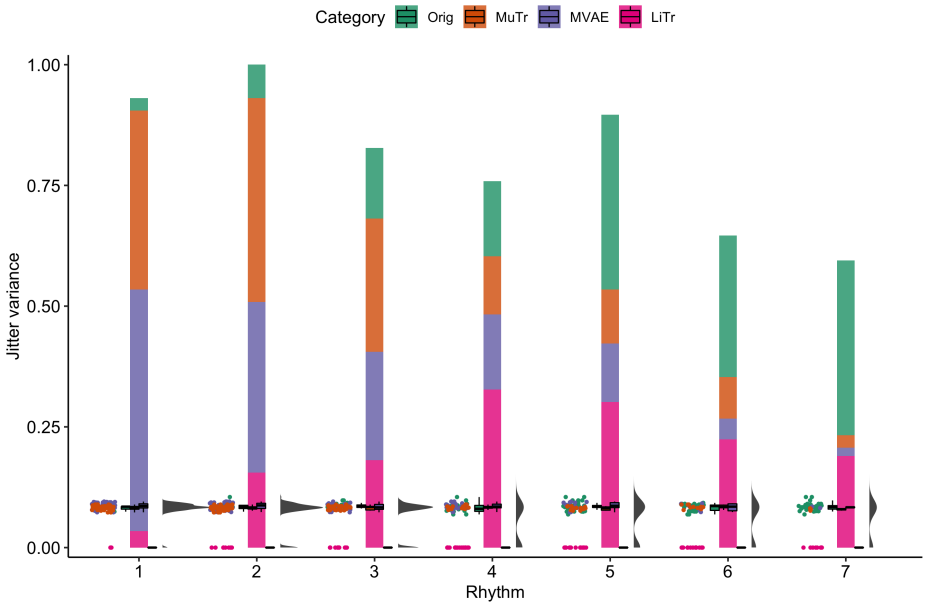


Figure B.156: Jitter variance against Rhythm ratings.

References

Kat Agres, Jamie Forth, and Geraint A Wiggins. Evaluation of musical creativity and musical metacreation systems. *Computers in Entertainment (CIE)*, 14(3):1–33, 2016.

Moray Allan and Christopher KI Williams. Harmonising chorales by probabilistic inference. *Advances in neural information processing systems*, 17: 25–32, 2005.

Micah Allen, Davide Poggiali, Kirstie Whitaker, Tom Rhys Marshall, and Rogier A Kievit. Raincloud plots: a multi-platform tool for robust data visualization. *Wellcome open research*, 4, 2019.

Teresa M Amabile. Social psychology of creativity: A consensual assessment technique. *Journal of personality and social psychology*, 43(5):997, 1982.

Charles Ames. The markov process as a compositional model: A survey and tutorial. *Leonardo*, 22(2):175–187, 1989.

Torsten Anders and Eduardo R Miranda. Constraint application with higher-order programming for modeling music theories. *Computer Music Journal*, 34(2):25–38, 2010.

Christopher Ariza. The interrogator as critic: The turing test and the evaluation of generative music systems. *Computer Music Journal*, 33(2):48–70, 2009.

Arthur Aron, Elliot J. Coups, and Elaine N. Aron. *Statistics for psychology*. Pearson, 6 edition, 2013.

Andreas Arzt, Sebastian Böck, and Gerhard Widmer. Fast identification of piece and score position via symbolic fingerprinting. In *ISMIR*, pages 433–438, 2012.

Jayme Garcia Arnal Barbedo. Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification. *Computers and electronics in agriculture*, 153:46–53, 2018.

Bernard Bel and Jim Kippen. *Bol Processor Grammars*, page 366–400. MIT Press, Cambridge, MA, USA, 1992. ISBN 0262521709.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 1*, pages 1171–1179, 2015.

Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade*, pages 437–478. Springer, 2012.

CH Bennett. Logical depth and physical complexity. In *A half-century survey on The Universal Turing Machine*, pages 227–257. 1988.

Margaret A. Boden. *The creative mind: Myths and mechanisms*. Weidenfield and Nicholson, 1990.

Margaret A Boden et al. *The creative mind: Myths and mechanisms*. Psychology Press, 2004.

Roberto Bresin and Giovanni Umberto Battel. Articulation strategies in expressive piano performance analysis of legato, staccato, and repeated notes in performances of the andante movement of mozart's sonata in g major (k 545). *Journal of New Music Research*, 29(3):211–224, 2000.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020.

Cambridge University Faculty of Music. *Music Tripos courses*. Cambridge, UK, 2010.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. *arXiv preprint arXiv:2012.07805*, 2020.

Filippo Carnovalini and Antonio Rodà. Computational creativity and music generation systems: An introduction to the state of the art. *Frontiers in Artificial Intelligence*, 3:14, 2020.

Carlos Eduardo Cancino Chacón and Maarten Grachten. The basis mixer: a computational romantic pianist. In *Late-Breaking Demos of the 17th International Society for Music Information Retrieval Conf.(ISMIR)*, 2016.

Harold Cohen. Colouring without seeing: a problem in machine creativity. *AISB Quarterly*, 102:26–35, 1999.

Thomas Edward Collins. *Improved methods for pattern discovery in music, with applications in automated stylistic composition*. PhD thesis, The Open University, 2011.

Tom Collins and Christian Coulon. MAIA Util: An NPM package for bridging web audio with music-theoretic concepts. In *Proceedings of the Web Audio Conference*, pages 47–52, Trondheim, Norway, 2019.

Tom Collins and Robin Laney. Computer-generated stylistic compositions with long-term repetitive and phrasal structure. *Journal of Creative Music Systems*, 1(2), 2017. doi: https://doi.org/10.5920/JCMS.2017.02.

Tom Collins, Jeremy Thurlow, Robin Laney, Alistair Willis, and Paul Garthwaite. A comparative evaluation of algorithms for discovering translational patterns in baroque keyboard works. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 3–8, 2010.

Tom Collins, Robin Laney, Alistair Willis, and Paul H Garthwaite. Chopin, mazurkas and markov: Making music in style with statistics. *Significance*, 8(4):154–159, 2011.

Tom Collins, Andreas Arzt, Sebastian Flossmann, and Gerhard Widmer. SIARCT-CFP: Improving precision and the discovery of inexact musical patterns in point-set representations. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 549–554, 2013.

Tom Collins, Sebastian Böck, Florian Krebs, and Gerhard Widmer. Bridging the audio-symbolic gap: The discovery of repeated note content directly from polyphonic music audio. In *53rd International Conference: Semantic Audio*. Audio Engineering Society, 2014.

Tom Collins, Andreas Arzt, Harald Frostel, and Gerhard Widmer. Using geometric symbolic fingerprinting to discover distinctive patterns in polyphonic music corpora. In *Computational Music Analysis*, pages 445–474. Springer, 2016a.

Tom Collins, Robin Laney, Alistair Willis, and Paul H Garthwaite. Developing and evaluating computational models of musical style. *AI EDAM*, 30 (1):16–43, 2016b.

Simon Colton, Alison Pease, Joseph Corneli, Michael Cook, and Teresa Llano. Assessing progress in building autonomously creative systems. In *ICCC*, pages 137–145. Ljubljana, 2014.

D. Conklin. Discovery of distinctive patterns in music. *Intelligent Data Analysis*, 14(5):547–554, 2010.

Darrell Conklin and Ian H Witten. Multiple viewpoint systems for music prediction. *Journal of New Music Research*, 24(1):51–73, 1995.

Deryck Cooke. The language of music. 1959.

David Cope. *Experiments in musical intelligence*, volume 12. AR editions, 1996.

David Cope. *Computer models of musical creativity*. MIT Press Cambridge, 2005.

James P Crutchfield. The calculi of emergence: computation, dynamics and induction. *Physica D: Nonlinear Phenomena*, 75(1-3):11–54, 1994.

Michael Scott Cuthbert, Christopher Ariza, and Lisa Friedland. Feature extraction and machine learning on symbolic music using the music21 toolkit. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 387–392, 2011.

Cedric De Boom, Stephanie Van Laere, Tim Verbelen, and Bart Dhoedt. Rhythm, chord and melody generation for lead sheets using recurrent neural networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 454–461. Springer, 2019.

RAFAEL DE CLERCQ. Aesthetic Pleasure Explained: De Clercq. *The Journal of Aesthetics and Art Criticism*, 77(2):121–132, 04 2019. ISSN 0021-8529. doi: 10.1111/jaac.12636. URL `https://doi.org/10.1111/jaac.12636`.

Douglas Dempster. Is there even a grammar of music? *Musicae Scientiae*, 2 (1):55–65, 1998. doi: 10.1177/102986499800200104.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

James M Dickey and BP Lientz. The weighted likelihood ratio, sharp hypotheses about chances, the order of a markov chain. *The Annals of Mathematical Statistics*, pages 214–226, 1970.

Zoltan Dienes. Using bayes to get the most out of non-significant results. *Frontiers in psychology*, 5:781, 2014.

Chris Donahue, Huanru Henry Mao, Yiting Ethan Li, Garrison W. Cottrell, and Julian McAuley. Lakhnes: Improving multi-instrumental music generation with cross-domain pre-training. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2019.

Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018.

Paul Doornbusch. Computer sound synthesis in 1951: The music of csirac. *Computer Music Journal*, 28(1):10–25, 2004.

Kemal Ebcioğlu. An expert system for harmonizing chorales in the style of js bach. *The Journal of Logic Programming*, 8(1-2):145–185, 1990.

Douglas Eck and Juergen Schmidhuber. Finding temporal structure in music: Blues improvisation with lstm recurrent networks. In *Proceedings of the 12th IEEE workshop on neural networks for signal processing*, pages 747–756. IEEE, 2002.

Tuomas Eerola and P Toiviainen. A method for comparative analysis of folk music based on musical feature extraction and neural networks. In *Proceedings of the VII International Symposium of Systematic and Comparative Musicology and the III International Conference on Cognitive Musicology. University of Jyväskylä*, 2001.

Arne Eigenfeldt and Philippe Pasquier. Realtime generation of harmonic progressions using controlled markov selection. In *Proceedings of ICCC-X-Computational Creativity Conference*, pages 16–25, 2010.

Ahmed Elgammal, Bingchen Liu, Mohamed Elhoseiny, and Marian Mazzone. Can: Creative adversarial networks generating "art" by learning about styles and deviating from style norms. In *8th International Conference on Computational Creativity, ICCC 2017*. Georgia Institute of Technology, 2017.

Steven Feld and Aaron A Fox. Music and language. *Annual Review of Anthropology*, pages 25–53, 1994.

Sebastian Flossmann, Maarten Grachten, and Gerhard Widmer. Experimentally investigating the use of score features for computational models of expressive timing. In *Proceedings of the 10th international conference on music perception and cognition*, 2008.

Francesco Foscarin, Andrew McLeod, Philippe Rigaux, Florent Jacquemard, and Masahiko Sakai. ASAP: a dataset of aligned scores and performances for piano transcription. In *International Society for Music Information Retrieval Conference (ISMIR)*, pages 534–541, 2020.

Katrien Foubert, Tom Collins, and Jos De Backer. Impaired maintenance of interpersonal synchronization in musical improvisations of patients with borderline personality disorder. *Frontiers in psychology*, 8:537, 2017.

Peter Gardenfors. *Conceptual spaces: The geometry of thought*. MIT press, 2004.

Robert Gjerdingen. *A classic turn of phrase: Music and the psychology of convention*. University of Pennsylvania Press, Philadelphia, PA, 1988.

Robert Gjerdingen. *Music in the galant style*. Oxford University Press, Oxford, UK, 2007.

C. James Goodwin and Kerri A. Goodwin. *Research in psychology: Methods and design*. Wiley, New York, NY, 8th edition, 2016.

Maarten Grachten and Gerhard Widmer. Explaining musical expression as a mixture of basis functions. In *Proceedings of the 8th Sound and Music Computing Conference (SMC 2011)*, 2011.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.

Graham Grindlay and David Helmbold. Modeling, analyzing, and synthesizing expressive piano performance with graphical models. *Machine learning*, 65(2):361–387, 2006.

Gaëtan Hadjeres and Frank Nielsen. Anticipation-rnn: Enforcing unary constraints in sequence generation, with application to interactive music generation. *Neural Computing and Applications*, 32(4):995–1005, 2020.

Gaëtan Hadjeres, François Pachet, and Frank Nielsen. Deepbach: a steerable model for bach chorales generation. In *International Conference on Machine Learning*, pages 1362–1371. PMLR, 2017.

Curtis Hawthorne, Erich Elsen, Jialin Song, Adam Roberts, Ian Simon, Colin Raffel, Jesse Engel, Sageev Oore, and Douglas Eck. Onsets and frames: Dual-objective piano transcription. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2018.

Curtis Hawthorne, Andriy Stasyuk, Adam Roberts, Ian Simon, Cheng-Zhi Anna Huang, Sander Dieleman, Erich Elsen, Jesse Engel, and Douglas Eck. Enabling factorized piano music modeling and generation with the MAESTRO dataset. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=r1lYRjC9F7`.

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

Stephen A Hedges. Dice music in the eighteenth century. *Music & Letters*, 59(2):180–187, 1978.

Dorien Herremans and Elaine Chew. Morpheus: generating structured music with constrained patterns and tension. *IEEE Transactions on Affective Computing*, 10(4):510–523, 2017.

Kate Hevner. Experimental studies of the elements of expression in music. *The American Journal of Psychology*, 48(2):246–268, 1936.

Hermann Hild, Johannes Feulner, and Wolfram Menzel. Harmonet: A neural net for harmonizing chorales in the style of js bach. In *Proceedings of the*

4th International Conference on Neural Information Processing Systems, pages 267–274, 1991.

Lejaren A Hiller Jr and Leonard M Isaacson. Musical composition with a high speed digital computer. In *Audio Engineering Society Convention 9*. Audio Engineering Society, 1957.

Douglas Hofstadter. *Fluid Concepts and Creative Analogies: Computer Models of the Fundamental Mechanisms of Thought*. Basic Books, 1995.

John E Hopcroft, Rajeev Motwani, and Jeffrey D Ullman. Introduction to automata theory, languages, and computation. *Acm Sigact News*, 32(1): 60–65, 2001.

Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Ian Simon, Curtis Hawthorne, Noam Shazeer, Andrew M Dai, Matthew D Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer: Generating music with long-term structure. In *International Conference on Learning Representations*, 2018.

Berit Janssen, Tom Collins, and Iris Yuping Ren. Algorithmic ability to predict the musical future: Datasets and evaluation. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 208–215, 2019.

Geoffrey Jefferson. The mind of mechanical man. *British Medical Journal*, 1 (4616):1105, 1949.

Dasaem Jeong, Taegyun Kwon, Yoojin Kim, Kyogu Lee, and Juhan Nam. Virtuosonet: A hierarchical rnn-based system for modeling expressive piano performance. In *ISMIR*, pages 908–915, 2019a.

Dasaem Jeong, Taegyun Kwon, Yoojin Kim, and Juhan Nam. Graph neural network for music score data and modeling expressive piano performance. In *International Conference on Machine Learning*, pages 3060–3070. PMLR, 2019b.

Daniel D Johnson. Generating polyphonic music using tied parallel networks. In *International conference on evolutionary and biologically inspired music and art*, pages 128–143. Springer, 2017.

Anna Jordanous. A standardised procedure for evaluating creative systems: Computational creativity evaluation based on what it is to be creative. *Cognitive Computation*, 4(3):246–279, 2012.

Anna Jordanous. Evaluating evaluation: Assessing progress and practices in computational creativity research. In *Computational Creativity*, pages 211–236. Springer, 2019.

Anna Jordanous and Bill Keller. Modelling creativity: Identifying key components through a corpus-based approach. *PloS one*, 11(10):e0162959, 2016.

Michael Kennedy and Joyce Bourne. *The concise Oxford dictionary of music*. OUP Oxford, 2004.

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

Andrei N Kolmogorov. Three approaches to the quantitative definition of information. *Problems of information transmission*, 1(1):1–7, 1965.

Carol L Krumhansl. The psychological representation of musical pitch in a tonal context. *Cognitive psychology*, 11(3):346–374, 1979.

Carol L Krumhansl. *Cognitive foundations of musical pitch*. Oxford University Press, 2001.

Michael D Lee and Eric-Jan Wagenmakers. *Bayesian cognitive modeling: A practical course*. Cambridge university press, 2014.

David Lewin. *Generalized musical intervals and transformations*. Oxford University Press, USA, 2007. Originally published by Yale University Press, New Haven, 1987.

Feynman Liang. Bachbot: Automatic composition in the style of Bach chorales. Master's thesis, University of Cambridge, 2006.

Hyungui Lim, Seungyeon Rhyu, and Kyogu Lee. Chord generation from symbolic melody using blstm networks. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2017.

Tom Lodewyckx, Woojae Kim, Michael D Lee, Francis Tuerlinckx, Peter Kuppens, and Eric-Jan Wagenmakers. A tutorial on bayes factor estimation with the product space method. *Journal of Mathematical Psychology*, 55(5):331–347, 2011.

Ryan Louie, Andy Coenen, Cheng Zhi Huang, Michael Terry, and Carrie J Cai. Novice-ai music co-creation via ai-steering tools for deep generative models. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2020.

David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

Huanru Henry Mao, Taylor Shin, and Garrison Cottrell. Deepj: Style-specific music generation. In *2018 IEEE 12th International Conference on Semantic Computing (ICSC)*, pages 377–382. IEEE, 2018.

Max V Mathews, Joan E Miller, F Richard Moore, John R Pierce, and Jean-Claude Risset. *The technology of computer music*, volume 5. MIT press Cambridge, MA, 1969.

Cory McKay, Julie Cumming, and Ichiro Fujinaga. Jsymbolic 2.2: Extracting features from symbolic music for use in musicological and mir research. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 348–354, 2018.

Soroush Mehri, Kundan Kumar, Ishaan Gulrajani, Rithesh Kumar, Shubham Jain, Jose Sotelo, Aaron Courville, and Yoshua Bengio. Samplernn: An unconditional end-to-end neural audio generation model. In *5th International Conference on Learning Representations (ICLR 2017)*, 2017.

David Meredith. The computational representation of octave equivalence in the western staff notation system. In *In Cambridge Music Processing Colloquium*, 1999.

David Meredith. The ps13 pitch spelling algorithm. *Journal of New Music Research*, 35(2):121–159, 2006.

David Meredith, Kjell Lemström, and Geraint A Wiggins. Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music. *Journal of New Music Research*, 31(4):321–345, 2002.

Michael C Mozer. Neural network music composition by prediction: Exploring the benefits of psychoacoustic constraints and multi-scale processing. *Connection Science*, 6(2-3):247–280, 1994.

Daniel Müllensiefen and Marc Pendzich. Court decisions on music plagiarism and the predictive value of similarity algorithms. *Musicae Scientiae*, 13 (1_suppl):257–295, 2009.

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? In *Advances in Neural Information Processing Systems*, pages 4694–4703, 2019.

Loris Nanni, Yandre MG Costa, Alessandra Lumini, Moo Young Kim, and Seung Ryul Baek. Combining visual and acoustic features for music genre classification. *Expert Systems with Applications*, 45:108–117, 2016.

Adam Neely. Why the Katy Perry/Flame lawsuit makes no sense. `https://www.youtube.com/watch?v=0ytoUuO-qvg`, 2019. Accessed: 2020-10-30.

James R Norris and James Robert Norris. *Markov chains*. Number 2. Cambridge University Press, 1998.

Sageev Oore, Ian Simon, Sander Dieleman, Douglas Eck, and Karen Simonyan. This time with feeling: Learning expressive musical performance. *Neural Computing and Applications*, pages 1–13, 2018.

Ellen Otzen. Six seconds that shaped 1,500 songs. `https://www.bbc.co.uk/news/magazine-32087287`, 2015. Accessed: 2020-10-30.

Diarmuid P O'Donoghue. Statistical evaluation of process-centric computational creativity. In *Proceedings of the 4th International Joint Workshop on Computational Creativity*, 2007.

Niall O'Mahony, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Velasco Hernandez, Lenka Krpalkova, Daniel Riordan, and Joseph Walsh. Deep learning vs. traditional computer vision. In *Science and Information Conference*, pages 128–144. Springer, 2019.

François Pachet, Alexandre Papadopoulos, and Pierre Roy. Sampling variations of sequences for structured music generation. In *ISMIR*, pages 167–173, 2017.

François Pachet and Pierre Roy. Plagiarism risk detector and interface, 2020. US Patent application number US 2020/0372882 A1.

Aniruddh D Patel. Language, music, syntax and the brain. *Nature neuroscience*, 6(7):674–681, 2003.

Marcus Pearce and Geraint Wiggins. Towards a framework for the evaluation of machine compositions. In *Proceedings of the AISB'01 Symposium on Artificial Intelligence and Creativity in the Arts and Sciences*, pages 22–32. Citeseer, 2001.

Marcus Pearce, David Meredith, and Geraint Wiggins. Motivations and methodologies for automation of the compositional process. *Musicae Scientiae*, 6(2):119–147, 2002.

Marcus T Pearce and Geraint A Wiggins. Evaluating cognitive models of musical composition. In *Proceedings of the 4th international joint workshop on computational creativity*, pages 73–80. Goldsmiths, University of London, 2007.

Karl Popper. *The logic of scientific discovery*. Routledge, London, UK, 2005. Original work published 1959.

Karl Popper. *Conjectures and refutations: The growth of scientific knowledge*. Routledge, London, UK, 2014. Original work published 1963.

Donya Quick and Paul Hudak. Grammar-based automated music composition in haskell. In *Proceedings of the first ACM SIGPLAN workshop on Functional art, music, modeling & design*, pages 59–70, 2013.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Brigitte Rafael, Stefan Oertl, Michael Affenzeller, and Stefan Wagner. Using heuristic optimization for segmentation of symbolic music. In *International Conference on Computer Aided Systems Theory*, pages 641–648. Springer, 2009.

Diana Raffman. *Language, music, and mind.* The MIT Press, 1993.

Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. A hierarchical latent vector model for learning long-term structure in music. In *International Conference on Machine Learning*, pages 4364–4373, 2018.

Marcelo Rodríguez-López and Anja Volk. Symbolic segmentation: A corpus-based analysis of melodic phrases. In *International Symposium on Computer Music Multidisciplinary Research*, pages 548–557. Springer, 2013.

Charles Rosen. *The Classical Style: Haydn, Mozart, Beethoven*. WW Norton & Company, London, UK, 1997.

Jeffrey N Rouder, Paul L Speckman, Dongchu Sun, Richard D Morey, and Geoffrey Iverson. Bayesian t tests for accepting and rejecting the null hypothesis. *Psychonomic bulletin & review*, 16(2):225–237, 2009.

Alejandro Ruiz and Ian Simon. My only problem with Magenta's Transformer. Magenta Discuss Google Group, `https:`

//groups.google.com/a/tensorflow.org/g/magenta-discuss/c/Oxiq-Gdaavk/m/uHIsQZKtBwAJ, 2020. Accessed: 2020-10-30.

Norbert Schwarz. Self-reports: How the questions shape the answers. *American psychologist*, 54(2):93, 1999.

Eleanor Selfridge-Field. Conceptual and representational issues in melodic comparison. *Computing in Musicology*, 1999.

Cosma Rohilla Shalizi and Kristina Lisa Shalizi. Blind construction of optimal nonlinear recursive predictors for discrete sequences. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 504–511, 2004.

Dave Smith and Chet Wood. The'usi', or universal synthesizer interface. In *Audio Engineering Society Convention 70*. Audio Engineering Society, 1981.

Temple F. Smith and Michael S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195—197, 1981.

Iyar Stav. Musical plagiarism: a true challenge for the copyright law. *DePaul J. Art Tech. & Intell. Prop. L*, 25:1, 2014.

Anne Penfold Street and Deborah J Street. *Combinatorics of experimental design*. Oxford University Press, Inc., 1986.

Bob Sturm, Joao Felipe Santos, and Iryna Korshunova. Folk music style modelling by recurrent neural networks with long short term memory units. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2015.

Bob L Sturm and Oded Ben-Tal. Taking the models back to music practice: Evaluating generative transcription models built using deep learning. *Journal of Creative Music Systems*, 2:32–60, 2017.

Bob L Sturm, Oded Ben-Tal, Úna Monaghan, Nick Collins, Dorien Herremans, Elaine Chew, Gaëtan Hadjeres, Emmanuel Deruty, and François Pachet. Machine learning research that matters for music creation: A case study. *Journal of New Music Research*, 48(1):36–55, 2019a.

Bob LT Sturm, Maria Iglesias, Oded Ben-Tal, Marius Miron, and Emilia Gómez. Artificial intelligence and music: open questions of copyright law and engineering praxis. In *Arts*, volume 8, page 115. Multidisciplinary Digital Publishing Institute, 2019b.

Hao Hao Tan and Dorien Herremans. Music fadernets: Controllable music generation based on high-level features via low-level feature modelling. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2020.

John Thickstun, Zaid Harchaoui, Dean P Foster, and Sham M Kakade. Coupled recurrent models for polyphonic music composition. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2019.

Peter M Todd. A connectionist approach to algorithmic composition. *Computer Music Journal*, 13(4):27–43, 1989.

Ellis Paul Torrance. *Torrance tests of creative thinking: Norms-technical manual: Figural (streamlined) forms A & B*. Scholastic Testing Service, 1998.

Shubham Toshniwal, Anjuli Kannan, Chung-Cheng Chiu, Yonghui Wu, Tara N Sainath, and Karen Livescu. A comparison of techniques for language model integration in encoder-decoder speech recognition. In *2018 IEEE spoken language technology workshop (SLT)*, pages 369–375. IEEE, 2018.

Esko Ukkonen, Kjell Lemström, and Veli Mäkinen. Geometric algorithms for transposition invariant content-based music retrieval. In *Proceedings of the International Symposium on Music Information Retrieval*, pages 193–199, 2003.

Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. pages 125–125, 2016.

Johnny van Doorn, Alexander Ly, Maarten Marsman, and E-J Wagenmakers. Bayesian rank-based hypothesis testing for the rank sum test, the signed rank test, and spearman's ρ. *Journal of Applied Statistics*, pages 1–23, 2020.

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.

Anja Volk, Elaine Chew, Elizabeth Hellmuth Margulis, and Christina Anagnostopoulou. Music similarity: Concepts, cognition and computation. *J. New Music Research*, 45(3):207–209, 2016.

Eric-Jan Wagenmakers, Tom Lodewyckx, Himanshu Kuriyal, and Raoul Grasman. Bayesian hypothesis testing for psychologists: A tutorial on the savage–dickey method. *Cognitive psychology*, 60(3):158–189, 2010.

Avery Li-Chun Wang and Julius O. Smith III. System and methods for recognizing sound and music signals in high noise and distortion, 2012. Patent US 8,190,435 B2. Continuation of provisional application from 2000.

Xiaofeng Wang, Kemu Pang, Xiaorui Zhou, Yang Zhou, Lu Li, and Jianru Xue. A visual model-based perceptual image hash for content authentication. *IEEE Transactions on Information Forensics and Security*, 10(7): 1336–1349, 2015.

Gerhard Widmer. Machine discoveries: A few simple, robust local expression principles. *Journal of New Music Research*, 31(1):37–50, 2002.

Gerhard Widmer. Discovering simple rules in complex data: A meta-learning algorithm and some surprising musical discoveries. *Artificial Intelligence*, 146(2):129–148, 2003.

Gerhard Widmer. Getting closer to the essence of music: The con espressione manifesto. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(2):1–13, 2016.

Geraint A Wiggins. A preliminary framework for description, analysis and comparison of creative systems. *Knowledge-Based Systems*, 19(7):449–458, 2006.

Geraint A Wiggins. Computer models of musical creativity: A review of computer models of musical creativity by david cope. *Literary and Linguistic Computing*, 23(1):109–116, 2008.

Iannis Xenakis. *Formalized music: thought and mathematics in composition*. Number 6. Pendragon Press, 1992.

Li-Chia Yang and Alexander Lerch. On the evaluation of generative models in music. *Neural Computing and Applications*, 32(9):4773–4784, 2020.

Li-Chia Yang, Szu-Yu Chou, and Yi-Hsuan Yang. MidiNet: A convolutional generative adversarial network for symbolic-domain music generation. In

Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), pages 324–331, 2017.

Peter Yates. *Twentieth century music: Its evolution from the end of the harmonic era into the present era of sound*. Allen & Unwin, 1968.

Zongyu Yin, Federico Reuben, Susan Stepney, and Tom Collins. "A good algorithm does not steal–it imitates": The originality report as a means of measuring when a music generation algorithm copies too much. In *Artificial Intelligence in Music, Sound, Art and Design: 10th International Conference, EvoMUSART 2021, Held as Part of EvoStar 2021, Virtual Event, April 7–9, 2021, Proceedings 10*, pages 360–375. Springer International Publishing, 2021.

Zongyu Yin, Federico Reuben, Susan Stepney, and Tom Collins. Measuring when a music generation algorithm copies too much: The originality report, cardinality score, and symbolic fingerprinting by geometric hashing. *SN Computer Science*, 3(5):1–18, 2022.

Christoph Zauner. Implementation and benchmarking of perceptual image hash functions. Master's thesis, Upper Austria University of Applied Sciences, 2010.

Fan Zhang, Hongying Meng, and Maozhen Li. Emotion extraction and recognition from music. In *2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*, pages 1728–1733. IEEE, 2016.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657, 2015.