Visualization Research Center of the University of Stuttgart (VISUS)

Bachelorarbeit

Visual Comparison of Music Composing AI

Hannes Deichmann

Course of Study:	Softwaretechnik
Examiner:	Prof. Dr. Michael Sedlmair
Supervisor:	Simeon Rau, M.Sc., Frank Heyen, M.Sc.
Commenced:	April 26, 2023
Completed:	October 26, 2023

Abstract

In this thesis, we describe a tool that allows a user to visualize and abstract data from melodies generated by multiple AIs. Related work in the field of music composing with AI is investigated and background information for the tool is given. We introduce different use cases and problems that the tool is supposed to solve and consecutively we present different visualization approaches and glyphs used to display the data computed from the AI melodies. We evaluate our approaches and demonstrate usability and readability with exemplary case studies. An outlook and a perspective on future work are given, exploring different additional features and refinements that can be done.

Contents

List of Figures

List of Tables

1 Introduction

Composing is a complex and demanding task both for amateurs and professionals. This can be attributed to a complex theory that is challenging to grasp or to the significant role that creativity and inventiveness play in the process of composing. Creativity can slow down the process of composing since it is a skill that may not be created at any time. Sometimes it can even lead to being stuck for days because the composer is lacking in new ideas or doesn't know how to continue. With the recent advancements in AI, it has now become possible to use this new technique as an inspiration for new melodies or musical themes thus overcoming the problem of creative scarcity. Yet the technique can be overwhelming with its endless possibilities. An AI may produce hundreds of melodies in a second, but for a user, it is not an efficient way to go through every single one of them to find good melodies that satisfy his needs. Instead, data computing and visualization can help filter the melodies and obtain information faster.

It is also hard to get an objective viewpoint of what has already been composed and what the next steps should be to advance in the composition. As a way of easing the need for these skills and offering a solution, many researchers have tried to integrate AI into composing over the last years [CWBD20; HKN+20; REM+19; RHWS22; SSBK16]. They all try different approaches, from simple melody generation over filling gaps in melodies with fitting compositions to generating entire pieces.
However, it is still very difficult for users to understand how these AIs work. How do they retrieve their output or can we trust this AI are questions regarding the interaction between a composer and an AI. This is where visualization of data plays a major role, answering these questions and giving insight into how the output can be utilized best.

Showing why an AI succeeds in solving given problems or explaining why a given model must be preferred over another model is a key step in the interaction between a user and an AI. Visualizing these explanations is both an easy way to understand and a less bleak way of doing so.

There are many AIs [CWBD20; REM+19] that can be used by composers to help them during their work. However, the composer usually lacks knowledge of the model or its training data which leads to not knowing which model should be used. AIs are often designed to assist in resolving one specific field or problem and don't apply to every use case. This concept of a black box leaves the composer with one possibility to find a fitting AI, which is comparing the output of the different AIs. That means a model may be trained for example using pop songs thus generating melodies that can not be used by a classical composer [HKN+20], so the composer needs to filter these out and identify that this AI is not suitable.

Hence the goal of this work is to plan and implement a tool to compare different AI models and their generated melodies. Since understanding raw data can be difficult without background knowledge, the tool is supposed to visualize the different parameters and data retrieved from the melodies generated by a model. Focusing on a visual analytical approach, the tool aims at easy information retrieving and filtering, offering good usability and readability.

The tool should allow this not only for a one-to-one comparison but for many melodies by multiple AIs, visually comparing them simultaneously.

In Chapter 2 we discuss different work that researchers on the topic of AI composing have already done. Chapter 3 covers the basic information needed to understand the concept and idea behind the tool. Before we describe the different visualizations in chapter Chapter 5, we list the various use cases of the tool and how the different data needed for these use cases is computed in Chapter 4. We then evaluate examples of our implementations and the inferences they provide in Chapter 6. In the end, in Chapter 7, we will give a conclusion and an outlook on future work, discussing what other visualizations could be implemented or what features to refine.

2 Related Work

[TWM19] focuses on difficulty reading a piano roll or music in real-time. Since inexperienced composers or users might struggle to follow the music by reading a piano roll they designed and implemented a real-time visualization where only the notes played at the moment are visible. By fading out to the left they create an effect of flying to the right screen which corresponds to the music going ”forwards”. Additionally, the notes of each voice are colored differently making it easy to differentiate multiple voices. The study shows that coloring and highlighting enhance users' comprehension of melody movement.

Coloring is also used in [CKS10; MC07; Sav20] to visualize polyphonic piano rolls. Although in our tool only monophonic melodies are used, it still demonstrates the importance of color as a visualization tool.

To help beginner composers Huang et al. trained an AI to suggest new and more diverse chord progressions [HDG16]. They designed a simple tool for interacting with the user that allows him to enter a chord progression to start with after which he is shown a simple list of possible continuations. However, they do not analyze the progression or give any assistance in what chord achieves what emotion or feeling. This leaves users unsure of which chord to choose. The tool also does not explain why the suggested chords are fitting so the user simply has to trust the program. For our tool, We aim to provide visualizations of the parameters that affect our computed metrics, enabling the user to comprehend the tool's evaluations of melodies. Another tool designed by Rau et al. assists a user in his compositional process. An interface enables the user to generate a melody using AI and iteratively modify it to fit his needs [RHWS22].

Karimi et al. have shown that AI-based tools can indeed help with creative processes [KRS+20] and support users with developing their ideas faster resulting in an optimized workflow.

Huang et al. [HKN+20] researched the need for an efficient interaction between AI-powered tools and users. They observed the workflow of multiple groups when creating a song with the help of AI and found multiple aspects a good interface needs. Firstly, it should be divisible into multiple, smaller components so the analytical process is easier. Secondly, it should be controllable meaning the user can interact with the tool and focus on what information he wants to be displayed. Lastly, the interface needs to be comprehensible so trust in the AI is established. That way a good interaction between a human and an AI is possible. Keim et al. [KAF+08] have similar findings in their research on how visual analytics need to be conducted.

[WBK09] presents a new method of visualizing MIDI files, which are the most common way to digitally store melodic information. They can easily be converted into for example piano rolls or in this case by Wolkowicz et al. into 2D images visualizing the entire piece's structure or uncovering important themes.

[WHF03] proposes a tool addressed at beginners to help learn and understand a piece of music. With their tool, they offer a new way of visualizing a melody by simplifying the traditional staff notation into a single line that traces the overall pitch direction of the melody. Additionally, they display the amount of notes in a bar as a gray value in the background. As a result, the user can highlight and examine points of interest in the music. They show that new visualizations can lead to a different understanding and help in maintaining the user's interest [Fra20].

A similar approach was taken by [F13] in transforming a piano sheet music into a graph, where every note is transformed. The pitch is assigned a specific color and the length corresponds to the width of the bar. This transformation results in a high information-density graph that gives a good summary of an entire piece of music.

[CNP16] has developed a framework for visualizing and comparing songs. They introduce box plots and heatmaps to add additional information when viewing the melody. Computing their own complexity and stability values demonstrates that generating new data can assist new interpretations.

Using the key as a main analytical parameter is developed in [MC07] where a melody is split up into multiple parts. These parts are harmonically analyzed and visualized as slices in a 2D net where every point represents a possible key. By using coloring assignment and size as the encoding for the key distribution the user can extract harmonic relations and contextual changes.

Many works have developed an interface for interacting with an AI [ARL+99; RHWS22], focusing on explaining and navigating the use of one single AI. Contrarily, our tool works with multiple AIs and aims to help a composer choose which one to use or prefer over another.

3 Background

In this chapter, we give a basic explanation of our tool regarding its goal, the targeted users, and the workflow we aim for.
The main navigation strategy for inspecting the melodies is a global-to-local strategy, allowing the user to get an overview of the sampled melodies before digging into them and viewing a few selected ones. In this way, the user can visually filter and select the area he wants to delve deeper into.

3.1 User

Our tool targets three main groups, professional composers, amateurs, and AI developers who want to explore the influence of parameters and other metrics on the generated output.

Professional Composer

A professional composer might excel in theoretical knowledge but might lack creativity or new ideas. This can occur from having worked for years in his field in which it is only natural that one creates fewer new thoughts over a long time. In this case, the tool can be used to explore new melodies and assist in interacting with AIs by showing relations between an AI and its style of composed melodies. The composer can quickly filter the generated melodies by looking at different visualizations of different attributes and based on this information conclude which melodies could fit in his composition.

Since he is experienced in musical theory and composing in itself, he is also able to articulate which features or parameters the melody needs to fit. He does not need to rely on the tool to analyze for example the emotions of the melodies and instead uses visualizations of more basic parameters to search for the ones he explicitly needs.

Amateur

On the contrary, an amateur might have a lot of spontaneity and may be eager to start but would be overwhelmed by complex musical concepts like keys and scales.

It is an easy way for him to use this tool as a first step to write new music where he doesn't have to worry too much about theory since the AIs can stay in a key without any specification made by the user or compose in an entirely new key. The tool then analyzes the key and displays it to the user so he simply needs to filter for the key he wants his melody to be in.

Parameter	Definition
Start	the time stamp at which a note starts
End	the time stamp at which a note ends
Pitch	the frequency of a note which is consistent over the time it occurs

Table 3.1: Basic parameters of a note

The tool can also analyze different metrics of the generated melodies like rhythmic complexity or emotion conveyed by a melody. This helps a beginner to simply choose the right melody with the desired effect, transferring the creative process to the AIs.

AI Developer

For this type of user, it is less important to get high-quality or ”good”musical ideas. An AI developer wants to analyze his algorithms and improve them, which is why it is so crucial for him to know what parameter influences an AI or what aspect should be improved on. When changing the parameters on his AI he needs to see what changed in the basic melodic attributes to evaluate the influence of his changes. A visual tool for comparing and analyzing melodic data satisfies these needs. This tool enables the developer to obtain objective and measurable results and view them. A developer also might be more interested in parameters and metrics that do not influence the melody's quality and instead aim to produce melodies for a specific genre and improve his AI for only this goal.

3.2 Data Structure

Our tool compares and analyzes multiple AIs simultaneously. For this purpose, it allows the user to import data by any AI as long as its output fits the correct format. The melodies need to be encoded using a JSON format containing various fields for parameters like temperature, the total length, or a description.
The melody itself is stored in an array in which every note is one element.

For the sake of simplicity, the analyzed melodies are strictly monophonic which means that at every point in time, only one note is present. This means necessarily, that no chord progressions or polyphonic relations are possible. Every note has three basic attributes as listed in Table 3.1. A pitch, a start time, and an end time are all stored as data/value pairs in their respective note object.

Every bar consists of 16 quantized steps which, given a four-four time, results in one quarter consisting of 4 steps. Accordingly, the shortest note is a sixteenth note consisting of only one step. This means it is impossible to depict sixteenth-note triplets, thirty-second notes, or other uneven distributions.
The start and end time stamps are measured in absolute values and not with respect to the bar they appear in which would be done normally in musical theory. Using the quantized steps as the unit allows for easier calculation of a note's duration since the start step can be subtracted from the end step. However, it can be more challenging to determine in which bar or measure a note appears.

```
"melodies": [
  {
    "notes": [
      {
        "pitch": 64,
        "quantizedEndStep": 8,
        "quantizedStartStep": 4
      },
      {
        "pitch": 64,
        "quantizedEndStep": 12,
        "quantizedStartStep": 8
      },
      {
        "pitch": 62,
        "quantizedEndStep": 16,
        "quantizedStartStep": 12
      },
      {
        "pitch": 62,
        "quantizedEndStep": 20,
        "quantizedStartStep": 16
      },
```


Figure 3.1: An example of a melody stored in a JSON file

The pitch is given in a range from 0 to 127. Every value corresponds to a note on the chromatic scale starting with a C in the sub-contra octave (or C0) which means the melody can be in the range of 12 and a half octaves up to a G11.

Since a melody consists of multiple notes most of the time, we can derive more parameters by combining the above-mentioned parameters of one note with another note, resulting in a variety of different parameters (Table 3.2).

Parameter	Definition
Duration of a note	subtracting the start point from the end point
Duration of a melody	the sum of the duration of every note and rest
Number of notes	The amount of notes in one bar or total
Interval	Two notes following after another form an interval, either descending, ascending or static
Range	The difference between the highest and lowest occurring pitch

Table 3.2: Data computed by combining parameters

4 Use Cases

In this chapter, we introduce and specify the various use cases of our tool. We describe the situations they occur in and what problem a user wants to solve when using the tool. While we focus on the use cases described in the following, it is important to note that there are many other use cases for the tool which are not addressed in the scope of this study.

4.1 Rhythm

A user might want to select an AI for its ability to create melodies with a specific rhythm or search for a melody that supports a specific emotion.
Emotions can be heavily influenced by rhythm and tempo [FFL16; LMBT10; RBB11]. Rhythm, however, can be impacted by the amount of notes and rests, their start time, or the occurrence of specific note values. Since the beats per minute (bpm) are always the same for all melodies, the tempo is stale, but it can be indirectly altered by the number of notes. For example, if a melody has 16 notes in one bar it is perceived as a faster rhythm than compared to a bar consisting only of one whole note, although the bpm is the same.
Therefore a user can rule out or find a lot of fitting melodies just by inspecting the rhythm. Since the rhythm alone provides a lot of data, we can divide and analyze parts of it and still get substantial information.

4.1.1 Distribution

The first and easiest thing to analyze is the distribution of note values. If a melody has a proportionally bigger share of long note values it is more likely that this melody emphasizes a slow pace and thus a calmer atmosphere [LMBT10]. In the same sense, a bigger portion of short note values can help with a sense of suspension or excitement.
Another contributor to the conveyed emotional effect is the amount of rests in a melody. More rests leave less room for notes resulting in an overall emptier character which is often associated with calmness or sadness. As a consequence of that, a user filtering for a happy melody aims for a low portion of rests and the opposite when filtering for a calm melody.

4.1.2 Complexity

The complexity of a rhythm is an extensive metric that includes many parameters and can play a big part in which emotions are perceived. Studies have shown that for example sadness and tenderness correlate to a firm and slow rhythm while a happy melody is supported by a complex

rhythm [LMBT10]. This is why having a good complexity metric can be very valuable. In our case, a complex rhythm, resulting in a high metric value, indicates a higher level of difficulty compared to a rhythm with a lower complexity value.

Computing a good metric for these use cases needs to take into account multiple parameters which all influence rhythmic complexity. The first step is counting the total amount of notes c_{notes}, since with every note the difficulty of a rhythm increases. Another factor is how many changes $c_{changes}$ of note values exist, meaning a note has a different length than the one before it. The parameter $c_{offbeat}$ counts every note that starts at an offbeat. There have been various attempts at formalizing and predicting offbeat or syncopation measures [Thu08; TT08; TT18] which have not fully achieved a reliable or generally accepted definition. This is why for this metric we simplify our definition. An offbeat is any note that occurs against the prevailing pulse. The analyzed melodies in our tool are written in a simple meter four-four time, so the main pulses are the time steps 0, 4, 8, and 12. Let d_{note} be the duration of a notes duration measured in quantized steps and t_{start} the absolute start step of the note we can thus describe the requirement for an offbeat note as the following:

Offbeat note: a note is considered offbeat when t_{start} mod d_{note} != 0

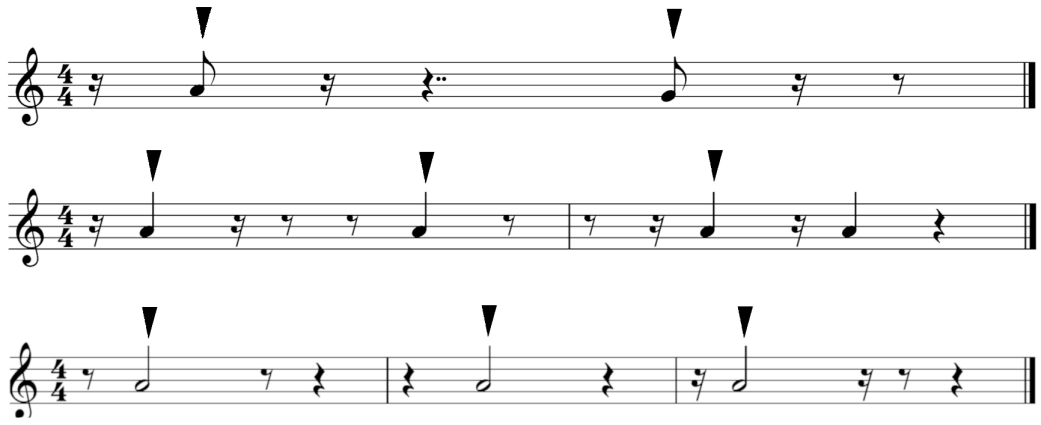


Figure 4.1: Offbeat notes marked with an arrow. The first line displays two possible offbeat eighth notes in a bar. Eighth notes are offbeat when starting on any time step which is not a multiple of 2. The second line shows offbeat quarter notes. The main difference compared to eighth is that they can start on any time step which is not a multiple of 4. The last line illustrates that half notes can be offbeat as well and even start on time steps which are multiples of 4.

Here the duration of a note is the deciding factor if it's counted as offbeat. It is worth mentioning that according to this definition, a sixteenth note is never counted as offbeat, since our smallest quantized step is a sixteenth note and as a result can never satisfy the requirement. This definition allows us to analyze every note independently from the ones before or after, simplifying the algorithm and computation of the complexity.

Rests have an influence as well in enhancing the complexity of a rhythm. Nevertheless, it is not the cumulative duration of rests within a melody that increases complexity, but rather the frequency of distinct rests. For instance, a whole-note rest may occupy 50 percent of a 2-bar melody, which should result in a diminished rhythmic complexity, despite its substantial temporal share. In contrast,

Value	Weight
𝅘𝅥𝅯 – 𝅘𝅥𝅰	5
𝄾 – 𝄿	3
𝅝 – 𝅗𝅥	1

Table 4.1: Rest to weight distribution

the occurrence of multiple sixteenth-note rests distributed between other notes is more likely to significantly increase the complexity. Therefore many short rests need to be more heavily weighted than a few long rests. As a first starting point we assign weights as described in Table 4.1 that can be refined in a future process. The sum of all weighted rests provides the parameter c_{rests}.

Additionally, while a change in consecutive note values may increase the difficulty of a rhythm, an offbeat start of a note might result in a more difficult rhythm which suggests the idea that the parameters need to be weighted. For testing purposes, we assumed a natural weighting where offbeat notes have the factor 10 and rhythmic changes have the factor 20. By dividing by 200 we map our function to a value between 0 and 1.

This yields the following equation for a metric describing the complexity of a rhythm:

$$Complexity = \frac{\sum c_{notes} + c_{changes} * 10 + c_{offbeat} * 20 + c_{rests}}{200}$$

4.2 Key

If the key plays a major role for the user he needs to differentiate the melodies and AIs by their key and if the notes that occur fit into this key.

For this problem, the tool first analyzes the key a melody is written in. Having established the key it is then possible to determine which notes in the melody are in the scale of that key or are outside of it. Analyzing every note of a melody in this way delivers a relationship between an AI and its ability to compose melodies in a specific key.

Thus the user can distinguish if one AI is more suitable for his needs of composing in one key. For example in a pop song in which tonality is very important and keys rarely change, the user wants to find an AI that supports this way of composing.

In another use case, a classical composer wants to get complex tonal melodies that fit the high rate of key changes and more diverse music. Thus the user wants to use an AI that composes melodies with atonal notes not belonging to the key they are written in.

4.3 Harmony

A more complex use case expands the former key problem. Notes can have a harmonic meaning and can be experienced as good sounding without belonging to the original key of the melody. Thus an AI might compose a fitting melody but would be analyzed by our tool as composing a strange melody since it contains notes outside its key.

This is why for this problem the analysis of the key has to be expanded to include harmonically fitting notes that might have a harmonic meaning thus making sense to use in a melody. This is plausibly harder to do since it requires a more complex analysis of notes. Looking at the note itself is not enough anymore to determine if it is harmonious and instead, the context in which the note appears has to be taken into consideration.

4.4 Emotional Effect

One of the arguably biggest effects a composer wants to achieve with his music is to trigger an emotional response [Coo59]. Multiple studies have researched these effects and what characteristics produce different kind of emotions [CFR04; JS01; LMBT10; SKC00]. Thus it is important to cover the use-case that a user can filter melodies by emotions or examine which spectrum a melody has. As for the program we need a list of parameters or features that relate to a certain emotion which we can then use to group the melodies. These features are derived from different studies and summarized to generate 4.2. [LMBT10] have proposed a rule system that supports various correlations. We have associated major keys with emotions linked to happiness and minor keys with emotions associated with sadness. Linked to happiness is also a fast tempo or rhythm, simple harmonies, a high average pitch as well as a wide pitch range, and an upward melody. On the opposite of the spectrum is sadness which is mostly triggered by melodies with a small quantity of notes, thus a slow rhythm, a low average pitch, and a small interval range. Also, the melody often has a downward direction, intervals are mostly pure and harmonies are simple. We decided to cover and analyze five basic emotions which are happiness, suspension, excitement, calmness, and sadness. Each one of these addresses a different range of emotions and thus results in good coverage of human emotions.

An entry in Table 4.2 represents a correlation between an emotion and a melodic feature. A plus sign in this case represents a positive correlation, meaning the feature has a high probability of being true or having a high value, thus triggering the specified emotion. For example, the first row depicts a high chance of a melody being in a major key, ïsMajorïs true, and stimulates the emotions of happiness and calmness. A minus sign represents a negative correlation, meaning the feature has a high probability of being false or having a low value. An example is the feature Äverage Intervall Height”which is often low when a melody is associated with sadness or calmness.
This feature list builds upon the data and metrics we have established and defined in the other sections of this chapter while introducing additional metrics.

The parameter for pure and impure intervals analyzes every interval in a melody and determines to which category it belongs. A prime, a fourth, a fifth and an octave are considered pure intervals which consecutively categorizes every other interval as impure.

	Happiness	Sadness	Excitement	Calmness	Suspense
isMajor	+	-		+	
isMinor	-	+		-	
Rhythm complexity			+	-	+
Amount of notes	+	-	+	-	
Melody Range	+	-	+	-	
Average Intervall Height	+	-	+	-	
Amount Impure Intervalls	-			-	+
Amount Pure Intervalls	+	+		+	
Average Pitch	+	-	+	-	
Upwards Melody			+		+
Downwards Melody		+		+	

Table 4.2: Feature list for emotions

The average pitch is calculated by measuring every interval i_j of a note j. Let c_m be the amount of notes in a melody we can calculate the average pitch with the equation $\frac{\sum_0^{c_m-1} i_j}{c_m-1}$.

To determine whether a melody's direction is upwards or downwards we compute how often a pitch of the next note goes up and how often it goes down. For this parameter the interval range is irrelevant, since a melody could start with an octave interval upwards and afterwards continue with only falling intervals and the melody should still be considered as a descending melody, regardless of the high increase at the beginning.

It is worth mentioning that a melody does not need to have every characteristic of the above-mentioned list and can sometimes even oppose some of them. It is enough that a majority of the features are satisfied for one emotion to be triggered.
Since some emotions share correlations of features, melodies can be perceived ambiguously. Thus melodies aren't strictly linked to one emotion but rather to a combination of different emotions.

5 Visualization

In this chapter, we propose different visualization approaches targeted at solving the aforementioned use cases and problems. Visualization is a key concept to understanding and evaluating extensive data. Good visualization abstracts data, focuses on comparing relevant points, and enhances the recognition of patterns. To ensure an efficient interaction with AI the user must build up trust. This is why visualization is so important since it not only helps get the desired melody it also helps the user to understand why one AI may be more suitable than a different one. Since we have confirmed that there exists a variety of possible users our tool must be understood by every type of user. This implies not using established and sometimes over-complex representations of music; instead one must rely on new ways of visualization or must establish simpler forms. In the following sections we describe the various graphs used to depict some or all of the information from an AI-generated melody that is used in the tool. Depicting every information of a melody can be tricky and overwhelming, which is why the graphs used to represent melodies in this tool often display one specific attribute or metric. This allows the user to highlight one particular piece of information about the melodies and thus makes it possible for him to draw conclusions based on the visualization.

5.1 Chromatic Roll

The chromatic roll is used as a simplification or variant of the well-known piano roll. It visualizes the pitch of every note by linking it to a color derived from a fixed scale of color and displaying the note as one bar. Hence it dismisses the height on the y-axis as a parameter and instead relies on the colouring to visualize the pitch of a note (Figure 5.1). Since every note with the same pitch has the same color it is fairly simple to see and very fast to determine which pitch occurs often in a melody. Besides, the x-axis represents the same data as in the piano roll, which is every note's start- and end-time.

Figure 5.1: The color scale used to depict the note's pitch.

For example in Figure 5.2 the first melody consists of only one pitch which is a G. Whether using the chromatic roll or the piano roll, in both cases, it is easy to deduct that the melody has the same pitch for every note. Looking at the piano roll, it can be seen as one flat line of notes in which all notes have the same height while for the chromatic roll, it can be seen that every note has the same color. However, the chromatic roll allows the user to also immediately see that the note's pitch is a g. This still applies even when the glyph is scaled down, whereas the piano roll can struggle to maintain its readability. That is why the chromatic roll is more suitable for use in a graph with lots of melodies compared to the piano roll.

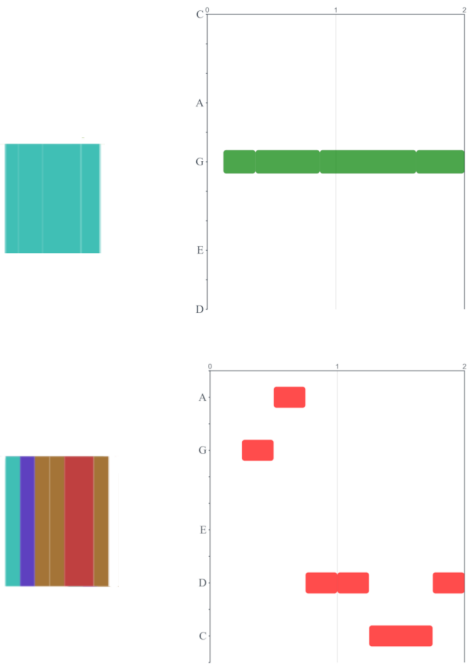


Figure 5.2: Chromatic rolls with their equivalent piano rolls

5.2 Rhythm Pie

The rhythm can be very important for a composer since it has a great impact on whether a melody is perceived as joyful or sad as described in Section 4.1.2. To visualize the distribution we implemented a pie graph. It is a classic pie chart in which every slice depicts a note value. The portion of the slice corresponds with the duration of all notes that have the same particular value. This allows the user to visualize if there are predominant note values or if there is a diverse distribution of note values. Also, the colors of the note values are derived from a red to blue color scale (Figure 5.3), meaning short values are depicted in a reddish color and long values in a bluish color.

Figure 5.3: The color scale used to depict the note values. Blue is used for a whole note and red for a sixteenth note.

5.3 Rhythm Complexity Scatterplot

Having computed our complexity metric as described in Section 4.1.2 we have one-dimensional data to display for every melody. Given a two-dimensional scatterplot where every melody is visualized by a simple data point, we can color in the rhythmic complexity using the same color scale as in Figure 5.3. However, in this case blue relates to a low complexity value whereas red symbolizes a high complexity value. This approach allows the user to quickly grasp which melodies have a difficult rhythm simply by observing the color.

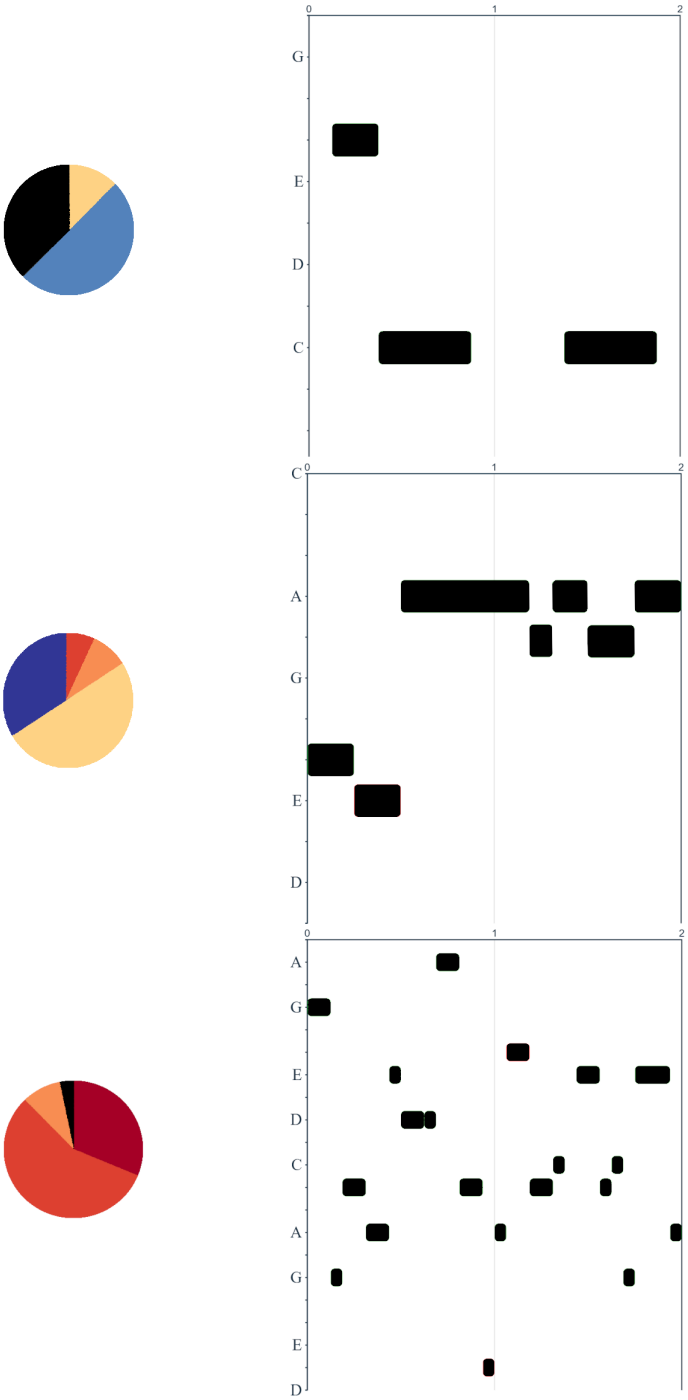


Figure 5.4: Rhythm pies with their corresponding piano rolls. The first rhythm pie has about one-third of rests, visualized by the black slice and overall long note values which relate to the color blue dominating. On the contrary, the third rhythm pie has a lot of short note values visualized by the big red slices.

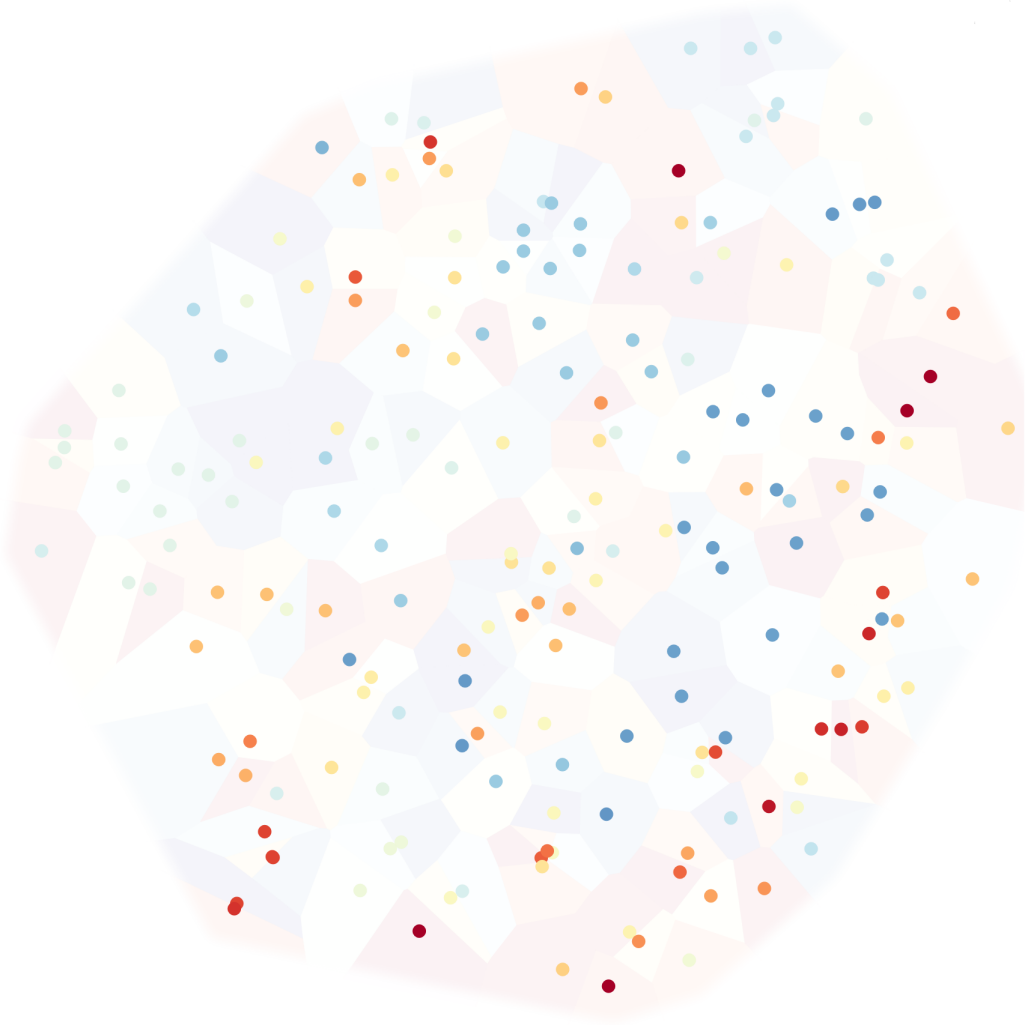


Figure 5.5: Melodies represented by a point in a 2D scatterplot and colored in their rhythmic complexity

5.4 Rhythm Complexity Highlighting

Adapting our complexity metric from Section 4.1.2 we can compute a bar complexity. This takes the same parameters into account but only from one beat and allows us to highlight where the difficult parts in a melody are. Here this is done by using a bar graph aligned with the x-axis of the piano roll so it is easy to see which beat complexity belongs to which beat of the melody. Figure 5.6 shows an example piano roll with a two-bar-long melody and the complexity values of each beat.

Here it can be seen that both times the beat 4 in each of the bars is the most complex beat since it contains quarter-notes and rests placed in difficult ways.

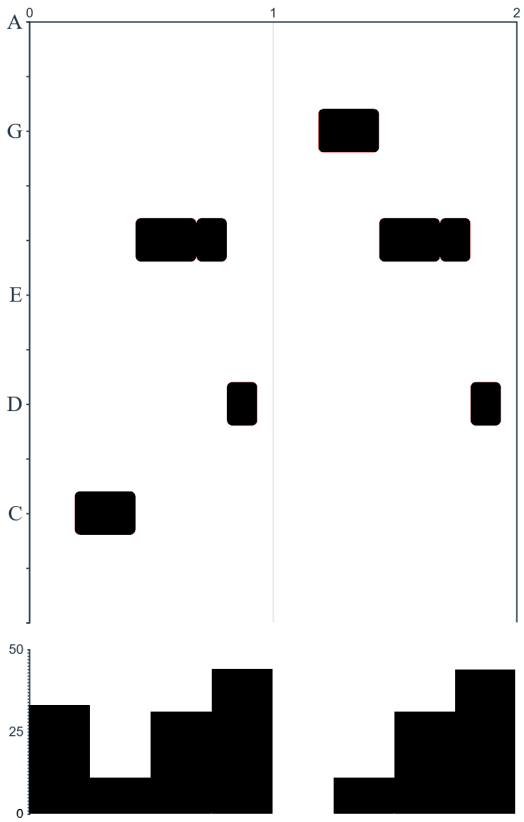


Figure 5.6: A piano roll with its highlighted rhythmic difficulties

5.5 Complexity Star Glyph

As written before, trust in a tool or AI is very important and can be supported by explaining the workflow and decisions made by the tool or AI. This is why we used a star glyph showing the user the various parameters that lead to a certain rhythmic complexity. Visualizing the values of these parameters can give fast insight into why a melody is evaluated by the tool as difficult or easy. Since we have four parameters, the star glyph needs four axes. The axis leading to the top of the star glyph presents the amount of rests. The axis to the right displays the number of offbeat notes while the left one shows the count of how often a note value changed. The axis to the bottom presents the total amount of notes in the melody.

As seen in Figure 5.7 a user can then extract the main factors of why a low or high complexity value has been computed by the program. Additionally, the star glyph is colored like in Section 5.3 so the complexity value can be seen simultaneously.

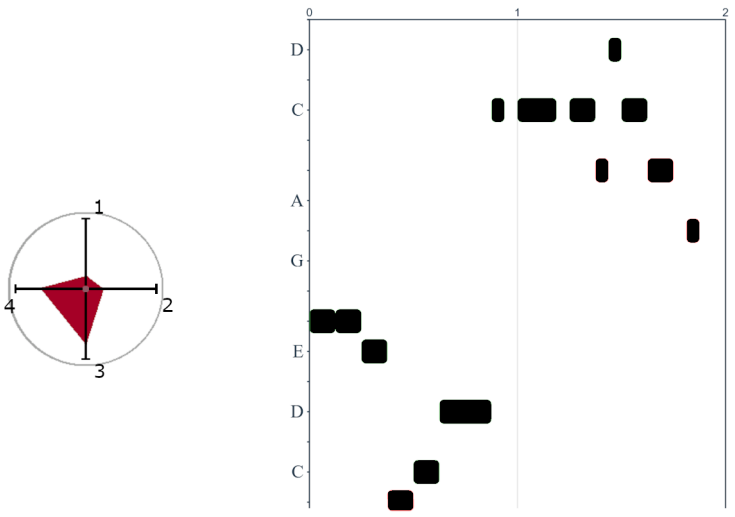


Figure 5.7: A star glyph showing each parameter of the melody's rhythmic complexity it represents. In this example the melody is complex as seen by the red color with almost no rests (axis 1), very few offbeat notes (axis 2), a lot of notes (axis 3), and a relatively high amount of note value changes (axis 4).

5.6 Emotion Clustering

We have described a list of features associated with certain emotions which we can now use to group the melodies of all AIs. We use dimensionality reduction to place our melodies in a 2D scatterplot according to our multidimensional feature list from Section 4.4. We use MDS and UMAP as dimensionality reduction algorithms to geometrically display our melodies in a way that correspondent melodies have a smaller distance than melodies that are different [Kru64; MHM20]. Figure 5.8 shows an example of how this visualization is put into practice.

This way of visualizing information allows for our tool to combine visualizations as it can be seen here. Each melody is placed according to their predicted emotion and, additionally, represented by a chromatic roll displaying different data. Utilizing this combination we can directly see clusters where one color is predominant

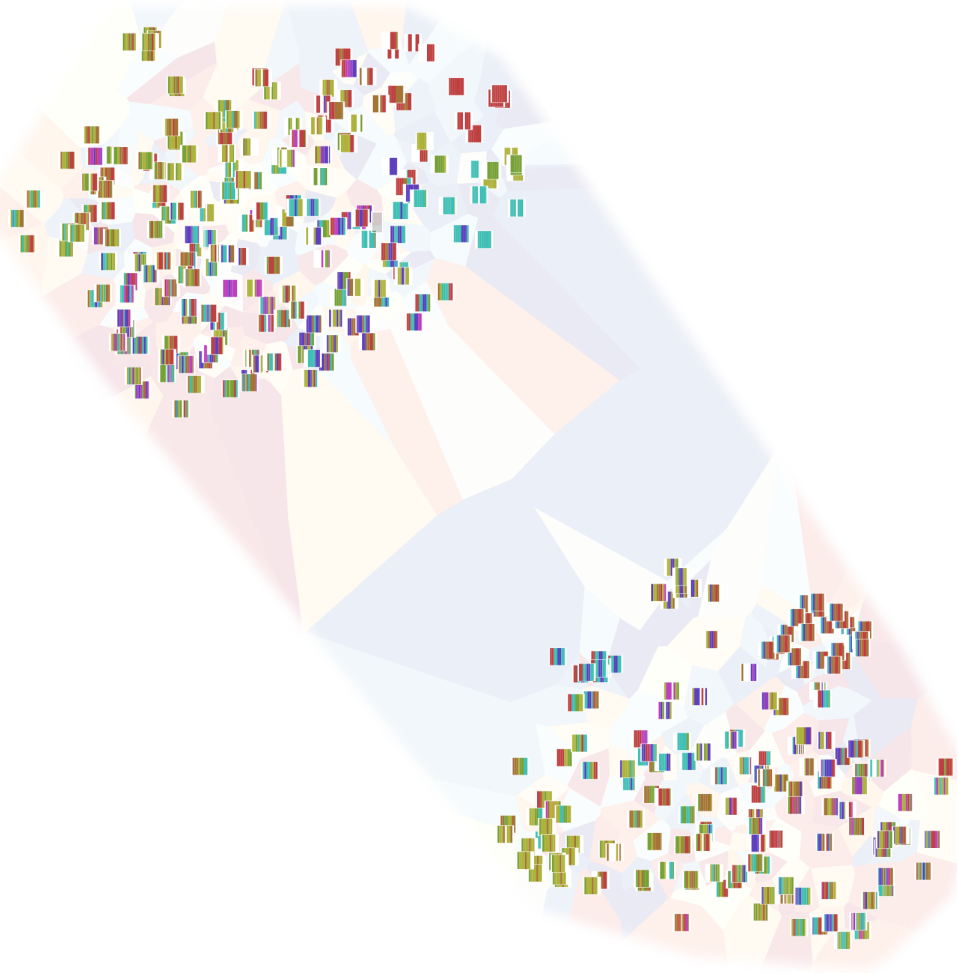


Figure 5.8: Melodies visualized in a scatterplot ordered by our feature list. Each melody is also represented as a chromatic roll

6 Evaluation

In this chapter, we evaluate our introduced features and show examples that allow us to draw conclusions and rate the AIs. We discuss visualizations taken from a real use case of our tool with melodies generated by ”Melody RNN”, a recurrent neural network developed by Google’s Magenta Project [Mag21]. The case studies in this chapter are examples of a series of steps a user might go through that also apply to models and data sets that are different than the ones we used.

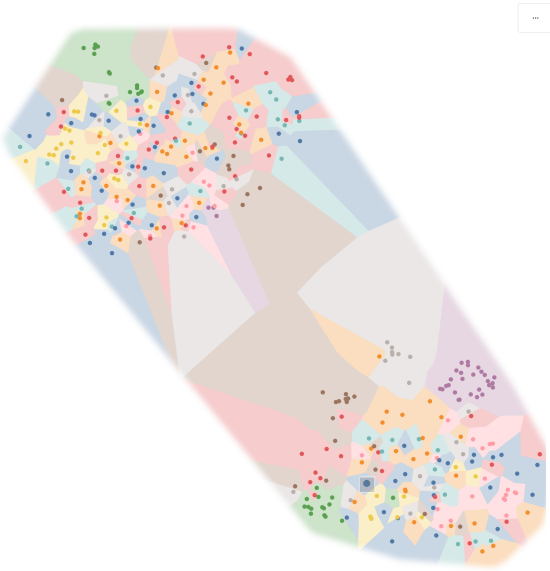
6.1 Finding an emotional reaction

In this case study we demonstrate how to find a melody with a specific emotional reaction. This approach is taken by composers who want an objective justification for why a certain melody should be chosen and need help finding said melody. An amateur can also use this approach to categorize the melodies so an evaluation of the melody’s emotional effect is easier. In this example, the user wants to find a melody that triggers suspense.

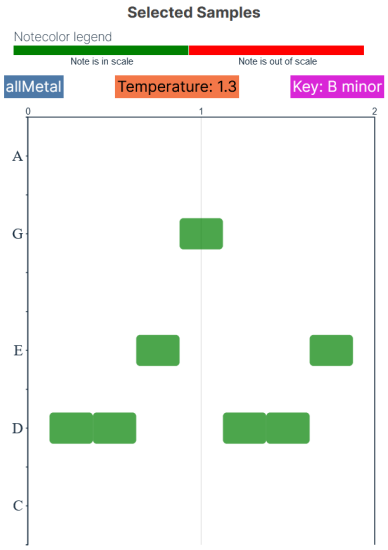
To achieve this, they use the emotion metric visualization where all melodies, represented by simple points, are grouped according to their emotion. This results in a scatterplot (Figure 6.1a) with two distinct halves, since the key of a melody separates the data in major and minor. Inspecting one melody of the bottom half in a detailed piano roll (Figure 6.1b) shows that the key is minor and thus this half is fitted for finding the desired melody. The rhythmic complexity is a deciding factor for distinguishing sadness from suspense thus the next step is to determine which melody has a high complexity. Switching the visualization so the point’s color correlates to the rhythmic complexity of its melody reveals a positive correlation between the complexity and the distance from the upper-left corner (Figure 6.1c). For that reason, melodies with a high rhythmic complexity are placed further away from the center. Thus the group of fitting melodies is further refined. Important for a suspenseful melody is a high amount of offbeat notes and rhythmic changes keeping the melody interesting, unpredictable and thereby suspenseful. Changing the encoding to star glyphs (Figure 6.1d) visualizes these two main factors on the left and right axes. Inspecting a melody with the correct distribution of rhythmic attributes also shows an upward direction of the melody which backs the selection of this melody as a fitting sample.

6.2 Differentiate using Rhythm Pies

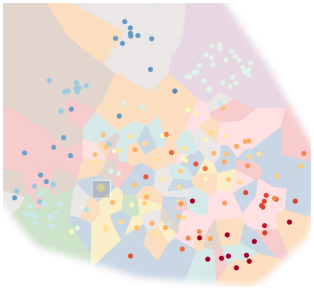
In this example we find a fitting melody by using rhythm pies. This time we want to find a melody with the associated emotion of happiness. As done in the example before the melodies are grouped by emotions and after establishing the bottom half as major the possible melodies are narrowed down. A happy melody has a decent rhythmic complexity and a relatively even distribution of



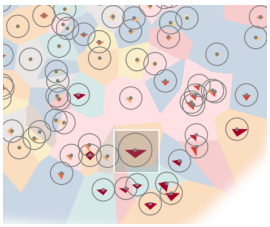
(a) Metric-based visualization using the metric for emotions. The background and point color represent the model that generated the melody. The visualization clearly separates sad melodies from happy and calm melodies because of the key distinction.



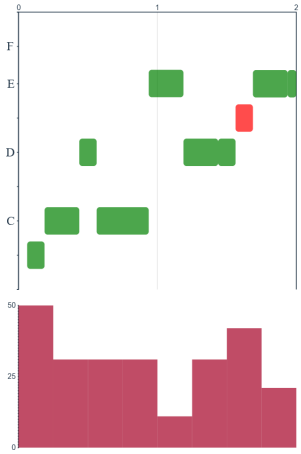
(b) The inspector of one the bottom half melodies shows the minor key B. However the rhythm and melody's direction do not fit the correct emotion.



(c) The point color is changed to represent the rhythmic complexity. This visualizes three groups of rhythmic complexity, where the complexity value increases from the upper-left corner to the bottom-right corner. Since the desired emotion is suspense, the focus needs to be melodies represented with a red point.



(d) Melodies represented with star glyphs. Selecting a fitting sample is simply done by examining the axes and choosing according values.



(e) The detailed piano roll sustains choosing this melody since the upward direction of the melody and the highlighted bar complexity fit the emotion of suspense.

Figure 6.1: Example steps for finding a melody associated with the emotion suspense

note values. Changing the visualization to rhythm pies instead of points allows the user to filter for these parameters. The correlation between a low rhythmic complexity and a high distance to the center visualized by the background colors suggests the selection of a melody in the middle. However, because of the clear information the rhythm pies provide, pies such as the one selected in Figure 6.2b can be ruled out simply because of the high amount of rests, visualized by the black slice. This is backed by the inspector which shows that the entire second bar is empty and therefore not triggering any emotion at all (Figure 6.2c). Instead selecting a pie with an even distribution of colors and slices results in a fitting melody (Figure 6.2d/e).

This evaluation shows that preattentive processing based on color is supported by the rhythm pies and allows for quick and systematic filtering without having to select and view every single melody sample.

6.3 Model Behavior

Combining the glyphs with a spatial representation of a model can reveal a lot about the behavior of a model and how it generates melodies. Placing melodies in a two-dimensional graph according to their emotional effect and the background color encoding the model leaves room for the data points, representing the samples, to encode additional information.

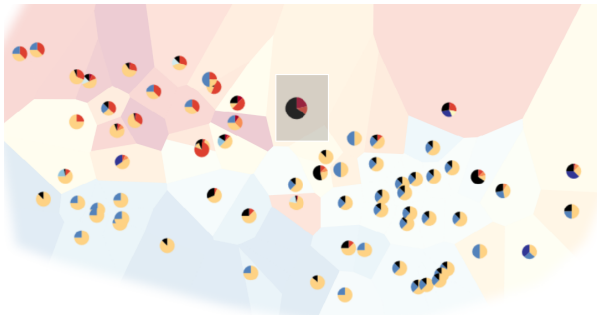
In Figure 6.3a we use chromatic rolls to draw a conclusion on how a model correlates with its use of pitch. Through examination of the top half, we can deduce that the brown, green, and purple models all use the same pitches when generating melodies. This leads to homogeneous melodies which are only slight variations of one another. Because of the geometrical position of these samples, we can also determine that the brown, green, and purple models generate melodies for a certain emotional effect.

On the contrary, glyphs that display a variety of different pitches and are generated by the same model allow for the conclusion that the model produces diverse variations of melodies. In Figure 6.3a we can assess that the red and orange model generate melodies with a high variation in pitch. As a result, the melodies are not as clearly grouped and affiliated with one emotion. Instead, the samples are displayed in a rather disjointed way.

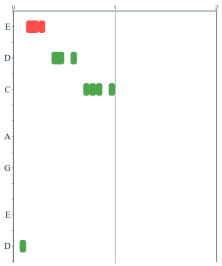
A similar conclusion can be drawn when inspecting chromatic rolls instead of rhythm pies (Figure 6.3b. The same models that proved to have only light variations in pitch are the same models that have the same rhythmic distribution for the majority of their melodies. Accordingly, the models with distinct variations in pitch are also producing samples with different rhythmic distributions.

This case study demonstrates the effective combination of visualizations to determine the behavior of different models. As a corollary, this allows for a better understanding of the AIs and thus helps in future iterations select an AI that provides the desired results the best.

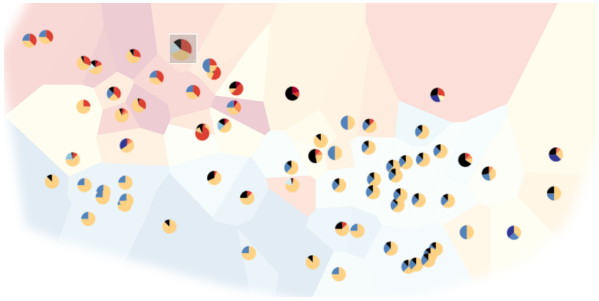
(a) The melodies are separated by key into two groups. For visual clarity both the background and point color visualize the rhythmic complexity.



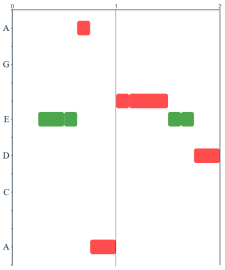
(b) Narrowed down group of melodies. A rhythm pie with a fitting value of rhythmic complexity is selected.



(c) The detailed view confirms what the rhythm pie has already shown which is that the melody has too many rests. Although solely by looking at the piano roll it is clear that the melody is only one bar long.

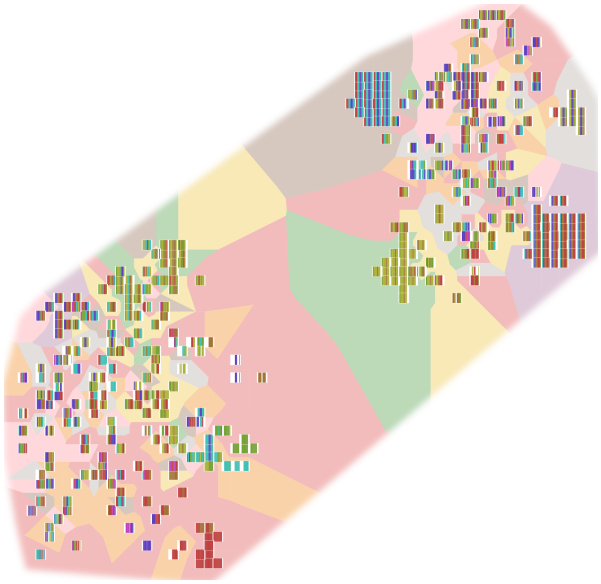


(d) Another rhythm pie with a relatively equal complexity is selected. Because of the distribution in the rhythm pie, it is evident that the rhythm is fitting for a happy melody.

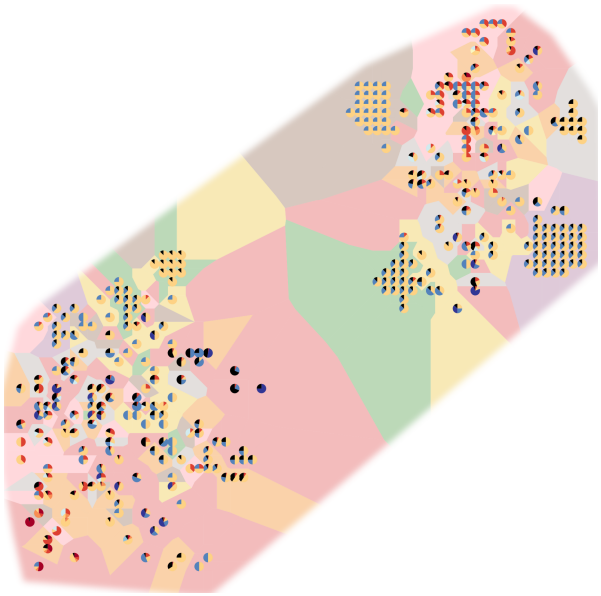


(e) The associated piano roll supports the finding that the melody satisfies the demanded emotion.

Figure 6.2: Utilizing rhythm pies to distinguish melodies



(a) The chromatic rolls visualize that some models (brown, purple, green) do not change a lot of notes whereas other models (red, orange) generate melodies with various different pitches.



(b) Changing to rhythm pies reveals that most models also rarely change or alter the distribution of note values in their generated melodies.

Figure 6.3: Analyzing model behavior by visualizing the melodies grouped by emotion

6.4 Interplay of Parameters

In this example, we show how different parameters of an AI interplay and influence each other. We order the melody samples by their temperature on the y-axis which ranges from 0.2 to 1.6. The x-axis relates to the number of notes in the sample. Representing the samples with star glyphs and the color of the star glyph encoding the rhythmic complexity, a positive correlation between temperature and rhythmic complexity can be derived (Figure 6.4. Furthermore, since we use star glyphs, we can determine the leading factor for the high complexity values. The star glyphs have increasing values on their left and right axes as the temperature increases. This demonstrates that an AI, when using a higher temperature, varies the start times of notes and their length, resulting in a high amount of rhythmic changes and offbeat notes. We can see that AIs do not tend to add pauses to create a variation which is shown by the star glyphs having no characterization on the top axis.

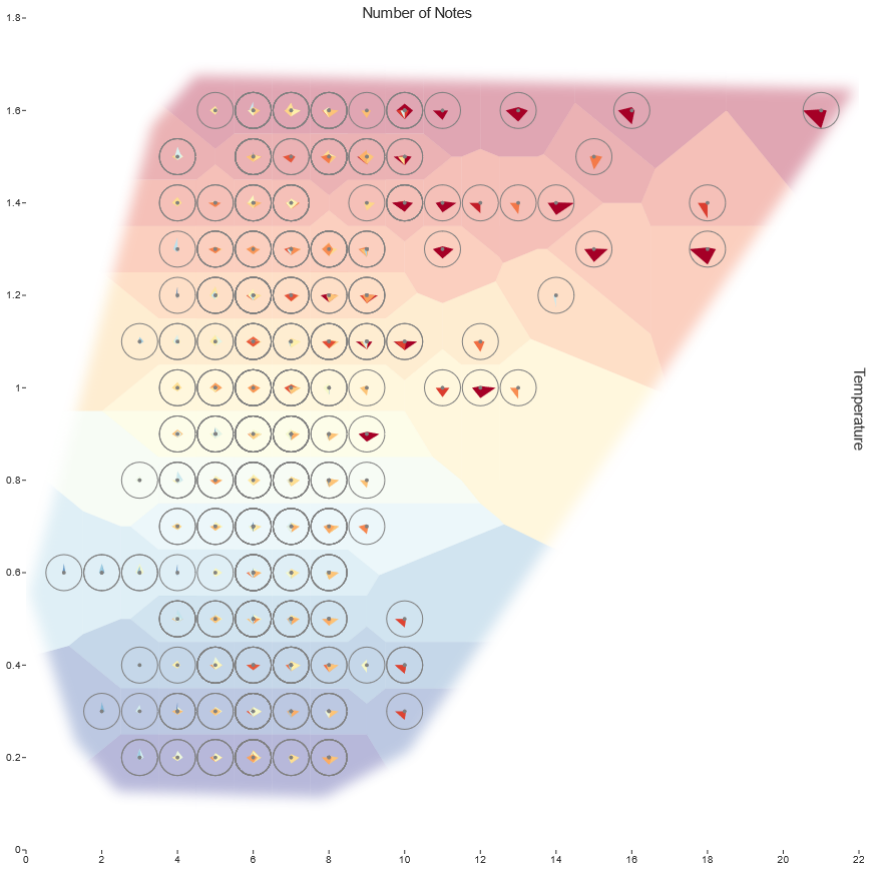


Figure 6.4: The melodies are aligned on the y-axis depicting the temperature used by the model to generate a sample. The background is colored accordingly from blue to red, representing the range from 0.2 to 1.6.

As shown by this example we can draw conclusions on the effect of AI parameters on the generated output and hence understand which parameters to change so the AI produces better samples.

7 Conclusion and Future Work

In this paper, we have listed different use cases for a tool that allows users to analyze and visualize multiple sets of melodies generated by different AIs. We proposed a variety of problems and use cases a user might encounter when filtering the melodies. In the visualization chapter, we introduced solutions for the formerly described use cases. The user has a palette of possible visualizations when examining the rhythm and its complexity like a rhythm pie, a bar graph, or a star glyph. We explored the different features of emotions a melody might trigger and implemented a visualization to display a set of melodies, grouped by their different emotion.
We showed different kinds of example cases utilizing our new visualizations and described the conclusions and implications that can be derived from them. We evaluated the described graphs and techniques by conducting different case studies, showing the usefulness of the new visualizations and revealing new ways of extracting information.

Future work needs to refine our already established metrics like rhythmic complexity. This means evaluating the weighting of the parameters and supporting them with a user-centered study. The study should focus on which parameters are perceived as the most challenging and thus should be weighted the most.
Rhythmic patterns are not a part of our complexity pattern. A complete complexity metric, however, needs to analyze the rhythm for reoccurring patterns, find similarities like slight variations, and adjust its value accordingly.

Clustering by emotions can also be explored more by improving the feature list used for the dimensionality reduction. Additional features based on patterns or context analysis can broaden the range of predictable emotions. At the moment our emotion clustering is based on data computed by looking at single notes instead of the context they appear in. Including the context would give a better representation of a piece. Broadening the range of emotions would improve and refine the classification and ultimately allow the tool to make better predictions.

The tool could be expanded to allow analysis of polyphonic music instead of only monophonic melodies. Polyphonics would open up a lot of possibilities for new metrics and criteria. New features could evaluate chord progressions, improve emotion prediction, or determine melodic climaxes and phrasing.

Since our data structure includes a 'bpm' attribute, the next step is to implement tempo changes in the melodies, allowing them to be played at different speeds. As it is now, a melody can only be slow or fast through the amount of notes. Adding tempo marks would allow for new interpretations of melodies that would have been uniform before.

A filter for genres could also help composers who try to create music in a specific genre. This would require a feature list for a specific genre, similar to our prediction of emotions. Changing the smallest quantized step size would also greatly improve the amount of possible melodies. Measuring the

length of a note in fractions would allow for uneven distributions like triplets to be used. Depending on the genre this change could be significant since some genres utilize frequently asymmetrical rhythms.

Bibliography

[ARL+99] G. Assayag, C. Rueda, M. Laurson, C. Agon, O. Delerue. "Computer-Assisted Composition at IRCAM: From PatchWork to OpenMusic". In: *Computer Music Journal* 23.3 (1999), pp. 59–72. DOI: 10.1162/014892699559896 (cit. on p. 14).

[CFR04] M. Costa, P. Fine, P. Ricci Bitti. "Interval Distributions, Mode, and Tonal Strength of Melodies as Predictors of Perceived Emotion". In: *Music Perception: An Interdisciplinary Journal* 22 (Sept. 2004), pp. 1–14. DOI: 10.1525/mp.2004.22.1.1 (cit. on p. 22).

[CKS10] P. Ciuha, B. Klemenc, F. Solina. "Visualization of Concurrent Tones in Music with Colours". In: MM '10. Firenze, Italy: Association for Computing Machinery, 2010, pp. 1677–1680. ISBN: 9781605589336. DOI: 10.1145/1873951.1874320. URL: https://doi.org/10.1145/1873951.1874320 (cit. on p. 13).

[CNP16] G. D. Cantareira, L. G. Nonato, F. V. Paulovich. "MoshViz: A Detail+Overview Approach to Visualize Music Elements". In: *IEEE Transactions on Multimedia* 18 (2016), pp. 2238–2246. URL: https://api.semanticscholar.org/CorpusID:31820427 (cit. on p. 14).

[Coo59] D. Cooke. *The Language of Music*. New York: Oxford University Press, 1959 (cit. on p. 22).

[CWBD20] K. Chen, C. Wang, T. Berg-Kirkpatrick, S. Dubnov. "Music SketchNet: Controllable Music Generation via Factorized Representations of Pitch and Rhythm". In: *CoRR* abs/2008.01291 (2020). arXiv: 2008.01291. URL: https://arxiv.org/abs/2008.01291 (cit. on p. 11).

[F13] H. F. *Sheet music visualization*. https://www.behance.net/gallery/6529923/Sheet-Music-Visualization. Accessed: 2023-10-18. 2013 (cit. on p. 14).

[FFL16] A. Fernández-Sotos, A. Fernández-Caballero, J. M. Latorre. "Influence of Tempo and Rhythmic Unit in Musical Emotion Regulation". In: *Frontiers in Computational Neuroscience* 10 (2016). ISSN: 1662-5188. DOI: 10.3389/fncom.2016.00080. URL: https://www.frontiersin.org/articles/10.3389/fncom.2016.00080 (cit. on p. 19).

[Fra20] M. S. Frank Heyen. "Supporting Music Education Through Visualizations of Midi Recording". In: *Posters IEEE Visualization Conf.* (2020) (cit. on p. 14).

[HDG16] C.-Z. A. Huang, D. Duvenaud, K. Z. Gajos. "ChordRipple: Recommending Chords to Help Novice Composers Go Beyond the Ordinary". In: *Proceedings of the 21st International Conference on Intelligent User Interfaces*. IUI '16. Sonoma, California, USA: Association for Computing Machinery, 2016, pp. 241–250. ISBN: 9781450341370. DOI: 10.1145/2856767.2856792. URL: https://doi.org/10.1145/2856767.2856792 (cit. on p. 13).

[HKN+20] C.-Z. A. Huang, H. V. Koops, E. Newton-Rex, M. Dinculescu, C. J. Cai. *AI Song Contest: Human-AI Co-Creation in Songwriting*. 2020. arXiv: 2010.05388 [cs.SD] (cit. on pp. 11, 13).

[JS01] P. Juslin, J. Sloboda. *Music and emotion, theory and research*. Jan. 2001. DOI: 10.1037/1528-3542.1.4.381 (cit. on p. 22).

[KAF+08] D. Keim, G. Andrienko, J.-D. Fekete, C. Görg, J. Kohlhammer, G. Melançon. "Visual Analytics: Definition, Process, and Challenges". In: (Mar. 2008). DOI: 10.1007/978-3-540-70956-5_7 (cit. on p. 13).

[KRS+20] P. Karimi, J. Rezwana, S. Siddiqui, M. Maher, N. Dehbozorgi. "Creative sketching partner: an analysis of human-AI co-creativity". In: Mar. 2020, pp. 221–230. DOI: 10.1145/3377325.3377522 (cit. on p. 13).

[Kru64] J. Kruskal. "Nonmetric multidimensional scaling: A numerical method". In: *Psychometrika* 29.2 (1964), pp. 115–129. URL: https://EconPapers.repec.org/RePEc:spr:psycho:v:29:y:1964:i:2:p:115-129 (cit. on p. 30).

[LMBT10] S. R. Livingstone, R. Mühlberger, A. R. Brown, W. F. Thompson. "Changing Musical Emotion: A Computational Rule System for Modifying Score and Performance". In: *Computer Music Journal* 34 (2010), pp. 41–64. URL: https://api.semanticscholar.org/CorpusID:17591825 (cit. on pp. 19, 20, 22).

[Mag21] MagentaProject. *Melody RNN*. https://github.com/magenta/magenta/blob/main/magenta/models/melody_rnn/README.md. [Online; accessed 21-October-2023]. 2021 (cit. on p. 33).

[MC07] A. Mardirossian, E. Chew. "Visualizing Music: Tonal Progressions and Distributions." In: Jan. 2007, pp. 189–194 (cit. on pp. 13, 14).

[MHM20] L. McInnes, J. Healy, J. Melville. *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*. 2020. arXiv: 1802.03426 [stat.ML] (cit. on p. 30).

[RBB11] D. Ramos, J. Bueno, E. Bigand. "Manipulating Greek musical modes and tempo affects perceived musical emotion in musicians and nonmusicians". In: *Brazilian journal of medical and biological research = Revista brasileira de pesquisas médicas e biológicas / Sociedade Brasileira de Biofísica ... [et al.]* 44 (Feb. 2011), pp. 165–72. DOI: 10.1590/S0100-879X2010007500148 (cit. on p. 19).

[REM+19] A. Roberts, J. Engel, Y. Mann, J. Gillick, C. Kayacik, S. Nørly, M. Dinculescu, C. Radebaugh, C. Hawthorne, D. Eck. "Magenta Studio: Augmenting Creativity with Deep Learning in Ableton Live". In: *Proceedings of the International Workshop on Musical Metacreation (MUME)*. 2019. URL: http://musicalmetacreation.org/buddydrive/file/mume_2019_paper_2/ (cit. on p. 11).

[RHWS22] S. Rau, F. Heyen, S. Wagner, M. Sedlmair. "Visualization for AI-Assisted Composing". In: *Proceedings of the 23rd International Society for Music Information Retrieval Conference (ISMIR)* (Bengaluru, India). ISMIR, Dec. 2022, pp. 151–159. DOI: 10.5281/zenodo.7316618. URL: https://doi.org/10.5281/zenodo.7316618 (cit. on pp. 11, 13, 14).

[Sav20] J. Savelsberg. "Visualizing the Structure of Music". In: (Jan. 2020) (cit. on p. 13).

[SKC00] E. Schellenberg, A. Krysciak, R. Campbell. “Perceiving Emotion in Melody: Interactive Effects of Pitch and Rhythm”. In: *Music Perception* 18 (Dec. 2000), pp. 155–171. DOI: 10.2307/40285907 (cit. on p. 22).

[SSBK16] B. L. Sturm, J. F. Santos, O. Ben-Tal, I. Korshunova. “Music transcription modelling and composition using deep learning”. In: *CoRR* abs/1604.08723 (2016). arXiv: 1604.08723. URL: http://arxiv.org/abs/1604.08723 (cit. on p. 11).

[Thu08] E. Thul. “Measuring the Complexity of Musical Rhythm”. PhD thesis. McGill University, June 2008 (cit. on p. 20).

[TT08] E. Thul, G. T. Toussaint. “On the Relation between Rhythm Complexity Measures and Human Rhythmic Performance”. In: *Proceedings of the 2008 C3S2E Conference*. C3S2E ’08. Montreal, Quebec, Canada: Association for Computing Machinery, 2008, pp. 199–204. ISBN: 9781605581019. DOI: 10.1145/1370256.1370289. URL: https://doi.org/10.1145/1370256.1370289 (cit. on p. 20).

[TT18] G. T. Toussaint, K. Trochidis. “On Measuring the Complexity of Musical Rhythm”. In: *2018 9th IEEE Annual Ubiquitous Computing, Electronics Mobile Communication Conference (UEMCON)*. 2018, pp. 753–757. DOI: 10.1109/UEMCON.2018.8796634 (cit. on p. 20).

[TWM19] M. Taenzer, B. C. Wünsche, S. Müller. “Analysis and Visualisation of Music”. In: *2019 International Conference on Electronics, Information, and Communication (ICEIC)* (2019), pp. 1–6. URL: https://api.semanticscholar.org/CorpusID:146118480 (cit. on p. 13).

[WBK09] J. Wolkowicz, S. Brooks, V. Keselj. “MIDIVIS: VISUALIZING MUSIC PIECES STRUCTURE VIA SIMILARITY MATRICES”. In: (Jan. 2009) (cit. on p. 13).

[WHF03] F. Watanabe, R. Hiraga, I. Fujishiro. “BRASS: Visualizing Scores for Assisting Music Learning”. In: *International Conference on Mathematics and Computing*. 2003. URL: https://api.semanticscholar.org/CorpusID:12915997 (cit. on p. 14).

All links were last followed on October 25, 2023.

Declaration

I hereby declare that the work presented in this thesis is entirely my own and that I did not use any other sources and references than the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations. Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before. The electronic copy is consistent with all submitted copies.

---

place, date, signature