Doctoral Thesis

Automatic generation of stage data for music games with an appropriate difficulty control method

Atsuhito Udo

Information Environment and Media Laboratory,
Graduate School of Information Science and Technology,
Hokkaido University

February, 2023

Abstract

The music game is one of the popular game styles where a player is typically requested to take specific actions at specific moments corresponding to the prominent acoustic epoch such as onsets in the music playing in the background. Stage data for such games record a sequence of actions (e.g., hitting a pad) the player has to take, and the timing of action is often represented by a moving target overlapping a stationary object on the screen. Creating such data is a difficult task that requires skilled artists. It is also time-consuming because such games require multiple sequences with different difficulty levels.

Multiple attempts at generating stage data using the machine learning approach have been made, including Donahue's[1] Dance Dance Convolution (DDC), which generates stage data for the music game called Dance Dance Revolution (DDR) from the user input of audio data of a piece of music and a desired difficulty level. DDC uses two models, step placement and step selection, to generate stage data. Step placement model finds onsets by employing convolutional neural network (CNN) and long-short term memory (LSTM), which step selection determines the type of action requirement (e.g., which button to press). It is found that when the desired difficulty level is lower, the generated stage data has significantly more action requirements (or targets) than the man-made counterpart.

We propose a method for generating sparse stage data with specified difficulty levels with two different approaches: 1. quantifying difficulty by examining the movement of the player required by the game 2. collecting statistics from man–made stage data analyzed from musical knowledge.

The difficulty of given stage data is mainly determined by the density of targets, but the complexity of the action also plays a role. Movement Cost (MC) is a value we propose which quantifies the difficulty of any stage data by calculating optimal action the player is required to make in order to play the game perfectly. By comparing the MC of the generated stage data and comparing it to the average of MC of the man-made ones, the two models can be modified to generate more or less targets or different type of targets.

On the other hand, introducing musical knowledge such as rhythm representations enables us to collect statistics from man-made stage data. This is because rhythm representations can be used to classify the targets. In the method we propose, the music is first divided into units called measures, then the measures are subdivided into certain number of sections. The onsets of the subdivisions are marked as significant points in the music, which is then used to classify the targets.

Using this method, we can create a preferred number of targets per class for each difficulty level, which we can reference to refine the generated stage data so that they have the level of difficulty desired by the user.

Contents

Chapter 1

Introduction

A video game is now a common entertainment and has become an indispensable element in our daily life. Different styles of video games have therefore been developed, such as shooter, action, simulation, etc. A music game is one of such popular game styles where a player is asked to complete a certain task indicated by a computer in correspondence with the music playing in the background. A typical music game (also called a rhythm game) requests the player to make some specific acitons, such as pushing buttons or hitting drum, at specific moments corresponding to the prominent acoustic epoch such as onsets of the background music and beats which are accents within a song that occur at a regular interval. The actions requested are often displayed as two objects, one moving and one stationary, on the screen. The timing of the action is expressed by overlapping those objects, and the type of the action by the shape, color, the different trajectory of the object, or any combination of those.

Preparing stage data, or a chart, is a difficult and time-consuming task. The charts are created by some skilled artists called charters, who have an ability to listen to the music and to pinpoint the onsets of each instrument involved in it to decide whether to place an action requirement at that moment or not. They are also requested to prepare multiple charts with different levels of difficulty (e.g., beginner, medium, and hard) for every music, making the task even more time-consuming.

This paper focuses on this problem for a particular game called Dance Dance Revolution (DDR), where players step on the four directional buttons on a sheet placed on the floor according to arrows scrolling up on the screen. The goal of our research is developing an automatic method for generating charts for this game with specified difficulty levels.

Chapter 2

Dance Dance Revolution

DDR is a Japanese music game developed by Konami in 1998. Figure 2.1 shows the components involved in DDR. The player is required to step on four pads on the floor with arrows on them (up, down, left, and right) according to the instruction given on the screen in front of the player. The arrows rise from the bottom toward the top of the screen. The player must make a step when arrows come to the specific positions at the top of the screen (indicated by grey arrows). Two arrows may appear at the same time, in which case the player must hit the pads on the same time. This usually causes the player to jump, so such a pattern is called a 'jump move.' Some arrows come with a 'trail,' in which case the player must keep the foot on the pad until the trail completely disappears off of the screen. Such arrows are called 'freeze arrows.' The timings when the arrows come to the top positions are in sync with the music playing in the background, and thus the player is required to pay close attention to the background music. The player scores better when he or she times the required steps more accurately.

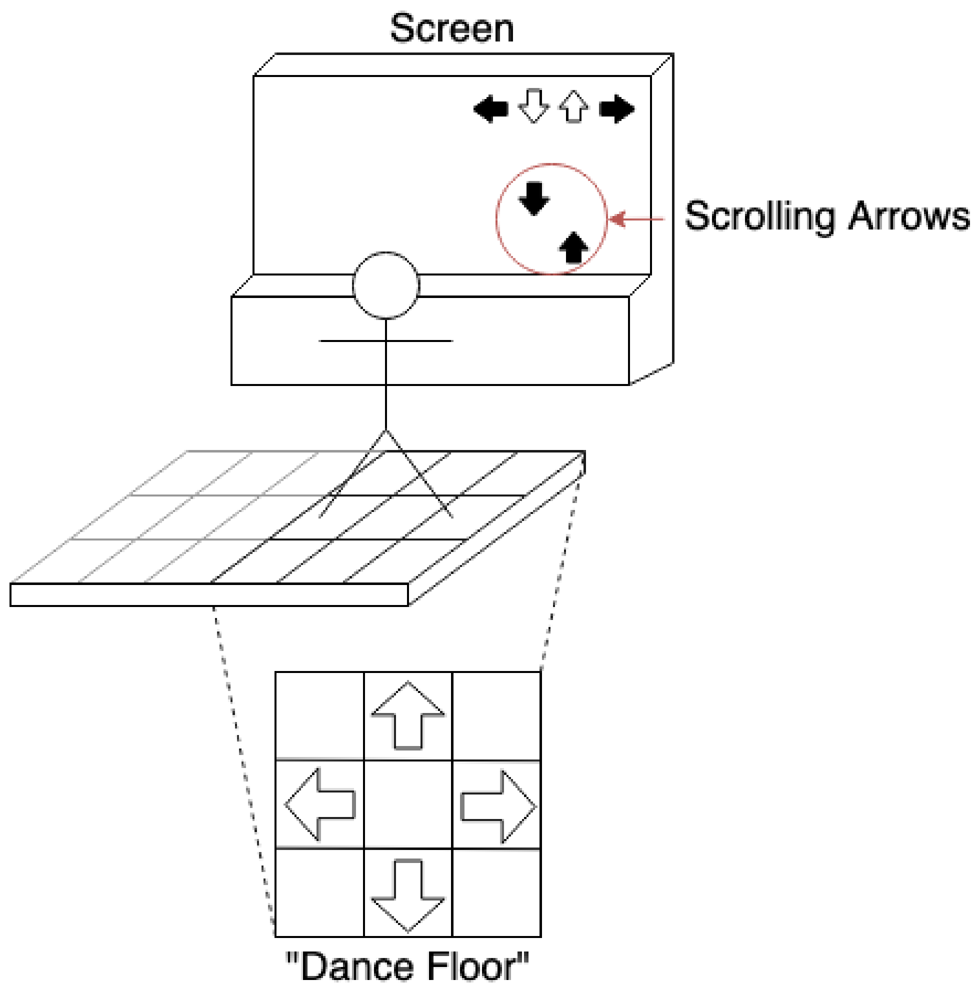

Fig. 2.1: Components of DDR.

Some versions of DDR and its clone simulator, StepMania, allow the player to create their own stage data to be played. In this paper, we simply refer to the arrows as targets when there is no need to specifically mention arrow's property.

Stage data for DDR include audio data and a set of five charts, which are different sets of targets presented to the player during their gameplay. Each chart is given a coarse level of difficulty by one of the five words, i.e., beginner, easy, medium, hard, and challenge, according to the relative density of the targets contained in it. Each chart also has an integer value that represents the difficulty across the game. The range of value can differ depending on the game, but for the most recent version of DDR, the range is 1 to 19, 19 being the most difficult rating for a chart. While it makes sense to use the level representation by word for comparing the level of difficulty of two charts for the same song, the integer level representation should be used to compare any given two charts. In this paper, we call the difficulty rating by a word (i.e., beginner, easy, ...) "coarse difficulty" while the difficulty rating by an integer (i.e., 1, 2, 3, ...) "fine difficulty."

DDR has a player community which uses simulators such as StepMania. StepMania allows the player to create charts for music they like, and thus a large dataset of music and chart has formed online. The community-crafted charts often have the difficulty rating ranging from 1 to 15, 15 being the most difficult rating. Such dataset can be used to generate charts for new songs by creating a model using machine learning technique.

Figure 2.2 shows the tool used in the man-made chart creation process in StepMania. The figure includes some of the following required information for a chart: 1. beat per minute, or BPM, of the audio, 2. the arrow map that defines the timings and the directions of arrows, and 3. a level of difficulty.

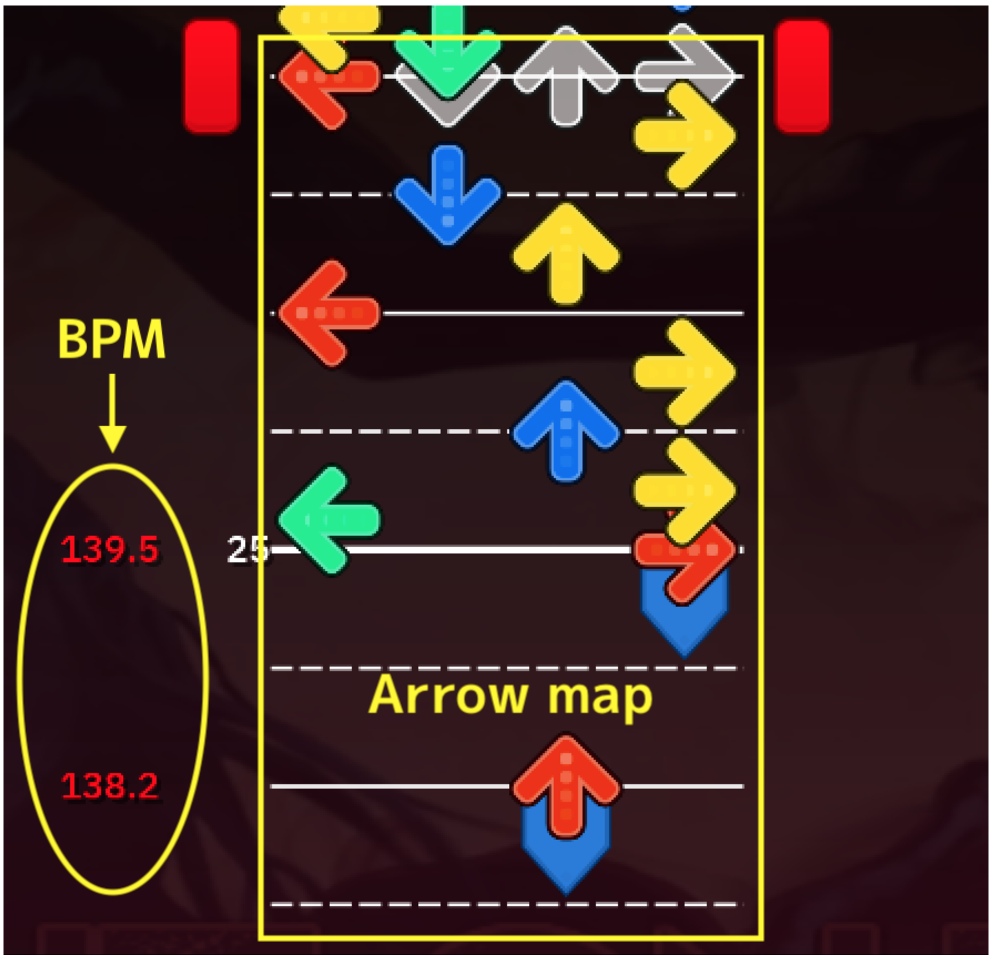

Fig. 2.2: Example of a chart in the StepMania editor.
BPM configurations and the arrow map can be seen.

Since the central aspect of the charts to be edited by charters is their targets, we also focus on modifying the property of targets, namely, their timing and the directions.

Chapter 3

Related Works

DDR has been the topic of several academic papers, including one by Hoysniemi [2], who investigated many aspects of DDR. While he did not mention automatic generation of the stage data, he conducted a survey about aspects in stage data that may be worth considering in the stage data generation process, including several idiom patterns in the directions of the arrows.

One important task in creating stage data is onset picking, which is one of the music information retrieval (MIR) tasks. Onset picking itself is an active topic in audio processing, and there are several groups that are attempting to detect the onset automatically. Zhou et al.[3] proposed an onset-picking method that utilized resonator time–frequency images [4] that generalize several audio signal analysis tasks into a single framework. They proposed two approaches to detecting onsets, pitch-based and energy-based. The performance of those approaches seem to vary depending on the nature of the audio (e.g., musical instruments). Schlüter and Böck [5] proposed a machine-learning approach to select musical onsets from audio with considerably high accuracy using an existing dataset of audio and the onset annotation. Donahue et al.[1] developed dance dance convolution (DDC), a system that generates stage data for DDR from any audio file containing music using Schlüter and Böck's method and natural language processing techniques to generate playable DDR stage data which looks similar to those made by human.

Similar approaches to improve the technique developed by Donahue et al. have been formulated in the past. Each approach focuses on different music games, and it is difficult to compare them in effectiveness directly since each music games have different charting philosophy, but they have some aspects that do not seem to cover.

Liang et al.[6] took different approach to label difficulty. They used density as a measure of difficulty instead of difficulty ratings given by humans, which may be subjective. This technique does not appear to consider the potential change in the perception of difficulty for two similarly dense patterns but have different musicology aspects. For example, a pattern consisting of four equally separate targets that are exactly on the beats is considered easier than the same target pattern offset by half a beat. This will be shown later in the results of our research to count the occurrences of the target in relation to the temporal position between beats. Halina and Guzdial[7] considered patterns formed with multiple targets in generating the charts. The problem with this approach is that the beginner charts are designed such that the target density is low; This low density may make it difficult for the player to recognize the patterns. Rasmus[8] mentioned the experiment conducted by Bolton[9] which revealed that two beats can be no further apart than 1600 ms (this limitation was later suggested to be 1800 ms by Fraisse[10]) for humans to recognize them as a rhythm. Our research shows that the average interval between targets is 1550 ms for the beginner charts, which is close to the said limit. Using patterns to generate charts may be effective for our purpose, but the human player may fail to notice the pattern generated by the method. Lin et al.[11] employed the concept called "beat phase" which is a similar concept to our method in that it uses musicology knowledge. Beat phase denotes the position of the target in relation to the temporal position between two beats. However, it is found, through our research, that the intervals between the targets for easier charts can be longer than the interval between the beats by as much as four times the same duration. Such long intervals cannot be accurately expressed by the beat phase.

These techniques may improve the ability to control the difficulty level of generated charts by suggesting an aspect of charts that represent their difficulty level. However, such aspects must represent the difficulty levels of the charts well enough to improve the performance of the technique, i.e., the aspect and the difficulty level should be linked as strongly as possible. In this paper, we investigate ways to represent the difficulty itself and the aspect of charts that represent their difficulty more effectively, apply them to DDC, and demonstrate its effectiveness.

DDC divides the chart creation process into two tasks: 1. finding the moment to place the targets 2. choosing the type of the targets (e.g., the direction of arrows). It has two machine-learned models, step placement model and step selection model to perform the respective tasks. The models have been trained by man-made charts that is available on the web, and they report better performance of their models when the charts created by the same charter is used as training data, whose total data size is 90 songs (210 minutes) and 450 charts. By learning to find the prominent epochs in the audio and to learn the sequence of target types that human charters would create, DDC succeeds in generating charts that have resemblance to what human charters would make. The system takes a PCM audio file and a desired difficulty of the chart by coarse difficulty.

Several experiments conducted by us reveal that DDC tends to generate charts that are more difficult than the user requests, especially when the requested coarse difficulty is on the easier side (i.e., beginner and easy), sometimes generating a chart that has 7 times more targets than the man-made ones. In this paper, we investigate this problem and propose potential methods to improve the difficulty control ability while retaining the ability to generate charts that are similar to man-made ones.

Chapter 4

Difficulty Control by Quantified "Difficulty"

In this chapter, we discuss two methods to improve the ability to control the level of difficulty by quantifying "difficulty" of given chart. Specifically, we focus on the fine difficulty rating and associate some value with it, then use such a value to evaluate the charts generated by the machine-learned model.

4.1 Quantifying "Difficulty"

As the fine difficulty rating of the charts increase, the density of the targets and the complexity of the sequence of directions of the targets also increase (Fig. 4.1). To quantify the difficulty and associate it with the fine difficulty rating, we introduce "movement cost," which is calculated by simulating an optimal foot movement to play the chart perfectly.

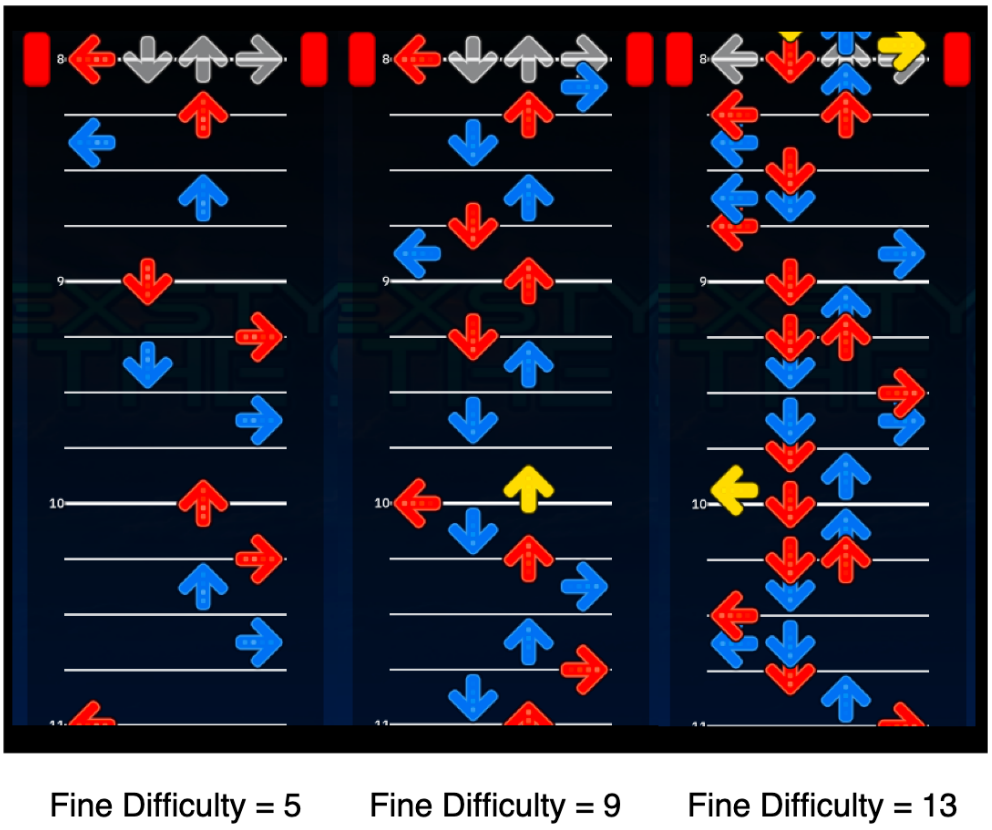

Fig. 4.1: Examples of man-made chart for some of the fine difficulty ratings.

4.1.1 DDR and optimal foot movement

Figure 4.2 shows the placement of the four pads the player must step on.

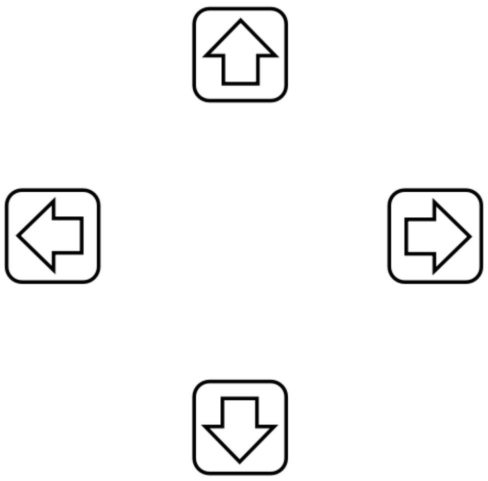

Fig. 4.2: The pad placement for DDR.

There are several points we can consider to calculate the optimal foot movement.

1. The player should place their two feet on two out of four pads.
2. Their feet may need to be moved from one pad to another to hit the targets.
3. It is beneficial for the player to use both feet to spread the strain on each foot.
4. The distance a foot needs to travel is not always the same, (e.g., Up to Right vs Up to Down).
5. For simplicity, the initial state of the feet are as follows: left foot on left pad, right foot on right pad.

To decide the optimal foot movements, we assume that both foot will be used alternately, unless:

1. the chart requires the player to hit the pad of the same direction multiple times, or
2. it is not possible (i.e., a jump move where two arrows appear at the same time or one foot must be kept on a pad because of 'freeze arrows.')

4.1.2 Movement cost

With these considerations in mind, we define movement cost M as follows:

$$M = \sum_{i=1}^{T} \frac{m_{t_i}}{t_i - t_{i-1}} \tag{4.1}$$

$$m_{t_i} = m_s(A_{t_i}) + m_d(A_{t_i}, A_{t_{i-1}}, \ldots, A_{t_1}) \tag{4.2}$$

where T is a number of all timings where arrows occur, t_i is the i-th timing since the beginning where arrows occur in seconds (we assume that t_0 is the beginning of the chart (or the song), hence $t_0 = 0$; for all the charts we used to calculate the movement cost, t_1 was sufficiently greater than zero), A_{t_i} represents all arrows that occur at t_i (A_{t_i} can include up to two arrows, in which case the player must jump to hit two pads simultaneously), $m_s(A_{t_i})$ is a function that returns the cost for arrows at a given timing (A_{t_i}), and $m_d(A_{t_i}, A_{t_{i-1}}, \ldots, A_{t_1})$ is a function that returns the distance between pads which a foot needs to move across to hit A_{t_i} from the positions of player's feet after hitting $A_{t_{i-1}}$ under the condition that the feet were moving according to the rules described at the beginning of section 4.1.2. This value is expected to reflect the distance and the velocity at which the foot needs to travel in order to hit the pads.

In this paper, we set $m_s(A_{t_i})$ to $1/16 \times$ (Number of arrows in A_{t_i}), and m_d as follows:

1. $m_d = 1$ if a foot moves to one panel to its neighbouring panel (e.g., Up to Right, Left to Down, Fig. 4.3–(a))
2. $m_d = \sqrt{2}$ if a foot moves to one panel to the panel on the opposite side (e.g., Left to Right, Fig. 4.3–(b))

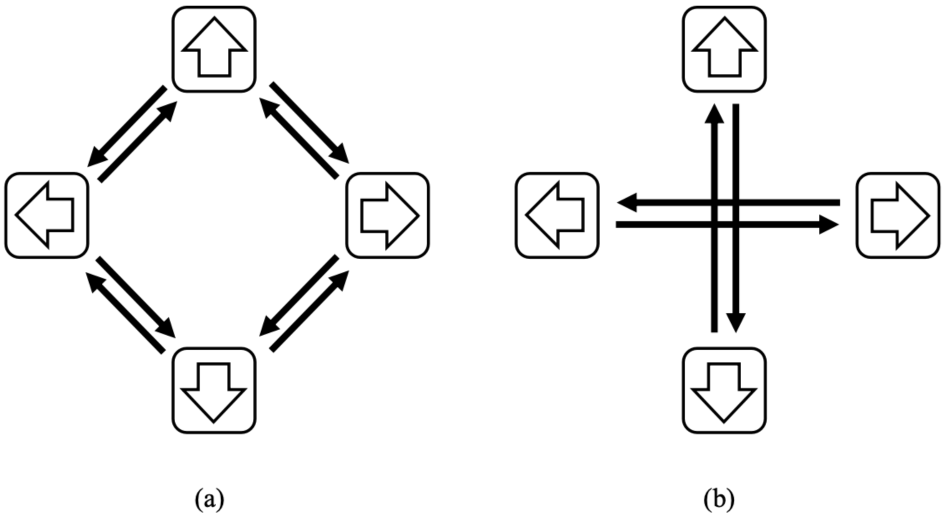

Fig. 4.3: Two possibilities of distance a foot may travel during game play.

We use an example chart shown in Figure 4.4 to demonstrate how Movement Cost is calculated. The top of the figure is the beginning of the chart, and the time flows from top to bottom. The horizontal lines indicate the onset of beats. Red arrows show that they occur at the same time that the line indicates while blue ones indicate that they occur at the time precisely between the moments indicated by two lines. If we assume that the horizontal lines are placed every 0.5 seconds and $t_1 = 1$, the movement cost for each arrow will be calculated as follows:

1. The left foot and the right foot placed on left and up pads respectively to set the initial state.

2. The first left arrow needs no foot movement, so $m_{t_1} = \frac{1/16}{1-0} = 0.0625$.

3. The second up arrow causes the right foot to move to the neighboring pad. $m_{t_2} = \frac{1/16+1}{1.25-1} = 4.25$.

4. The third down arrow causes the left foot to move to the neighboring pad. $m_{t_3} = \frac{1/16+1}{1.5-1.25} = 4.25$

5. The fourth right arrow causes the right foot to move to the neighboring pad. $m_{t_4} = \frac{1/16+1}{1.75-1.5} = 4.25$

6. The fifth up arrow causes the left foot to move to the pad on the opposite side. $m_{t_5} \frac{1/16+\sqrt{2}}{2-1.75} = 1/4 + 4\sqrt{2} \approx 5.906$

7. The final arrows make up a jump move, but the cost is calculated for each arrow. The left arrow causes the left leg to move to the neighboring pad. The right leg needs no movement. $m_{t_6} = \frac{(1/16+1)+(1/16)}{2.5-2} = 2.25$.

8. Finally, all the costs for the arrows are summed for the total movement cost of $M \approx 20.97$.

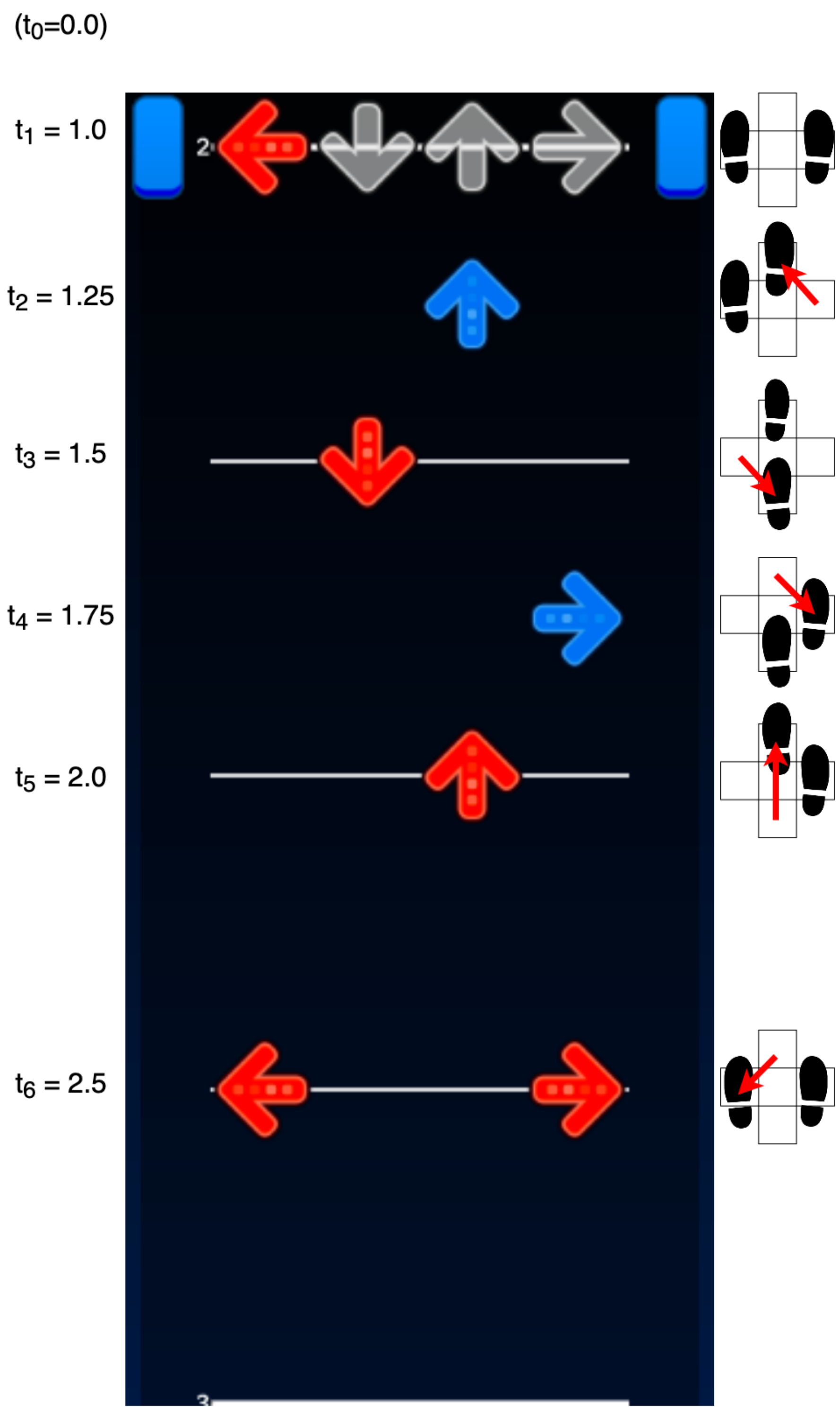

Fig. 4.4: Calculation of movement cost, visualized.
A white line represents a beat, which we assume to occur every 0.5 seconds in this example.

4.1.3 Experiment

An experiment to find the movement cost for each fine difficulty rating was conducted with $1,178$ man-made charts. First, the charts were classified by the fine difficulty rating given by the charter. For each rating, the movement cost of the chart with such a rating is calculated, and then divided by the length of the song in seconds to get the movement cost per second.

Figure 4.5 shows the result. The error bar shows the 95% confidence interval. The graph shows that the cost and the fine difficulty rating has positive correlation.

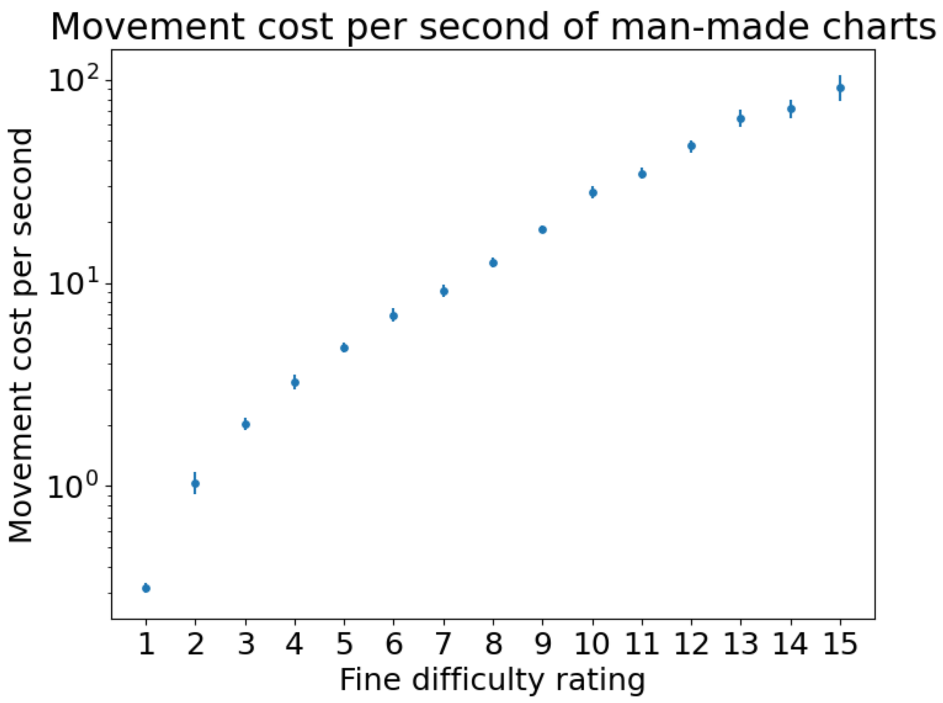

Fig. 4.5: Movement cost per second for each fine difficulty rating.

The same procedure was used to find the movement cost of the charts generated by DDC. Since DDC only supports specifying the coarse difficulty rating of the chart, we generate 10 charts for each coarse difficulty rating for total of 50 charts using songs that is not in the training data, but has the similar genre to the music in the training data.

Figure 4.6 shows the result of this experiment.

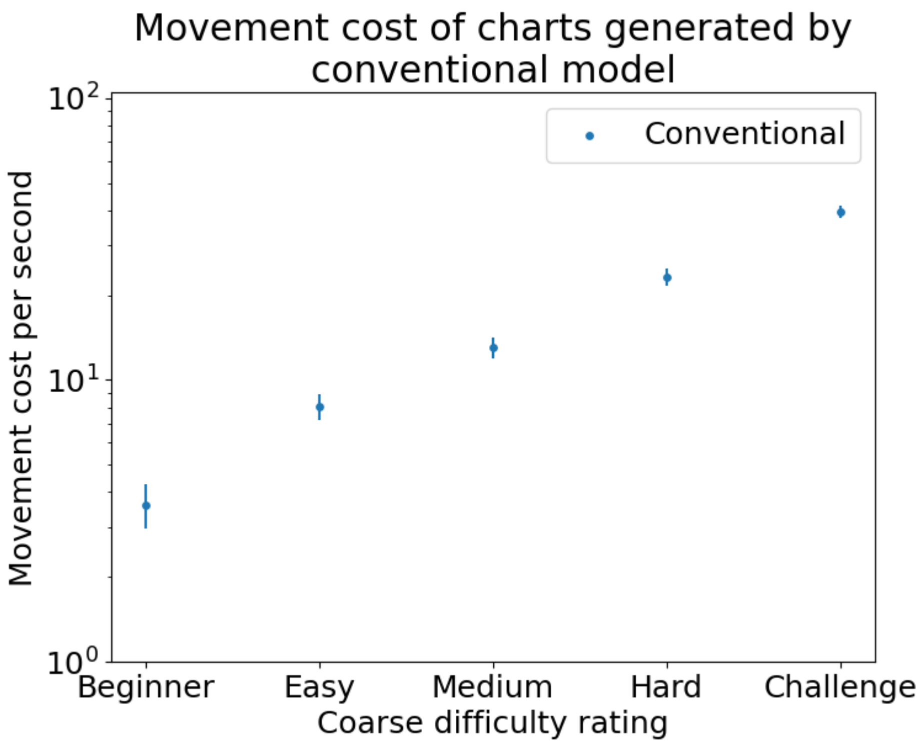

Fig. 4.6: Movement cost per second for each coarse difficulty rating choice that DDC offers.

Charts generated by DDC have fixed difficulty ratings for each coarse difficulty ratings before generating them. If we assume that DDC intends to generate charts with these fine difficulty ratings, we can compare the movement cost of the generated charts against that of man-made charts. Figure 4.7 and Table 4.1 shows the result of the comparison. The comparison reveals that the generated charts have higher movement cost than that of the man-made charts with the same fine difficulty rating. We have compared the movement cost of the generated charts (Figure 4.6) with that of the man-made charts (Figure 4.5), and found the actual fine difficulty ratings which, had human charters rated them, might have been found on such generated charts.

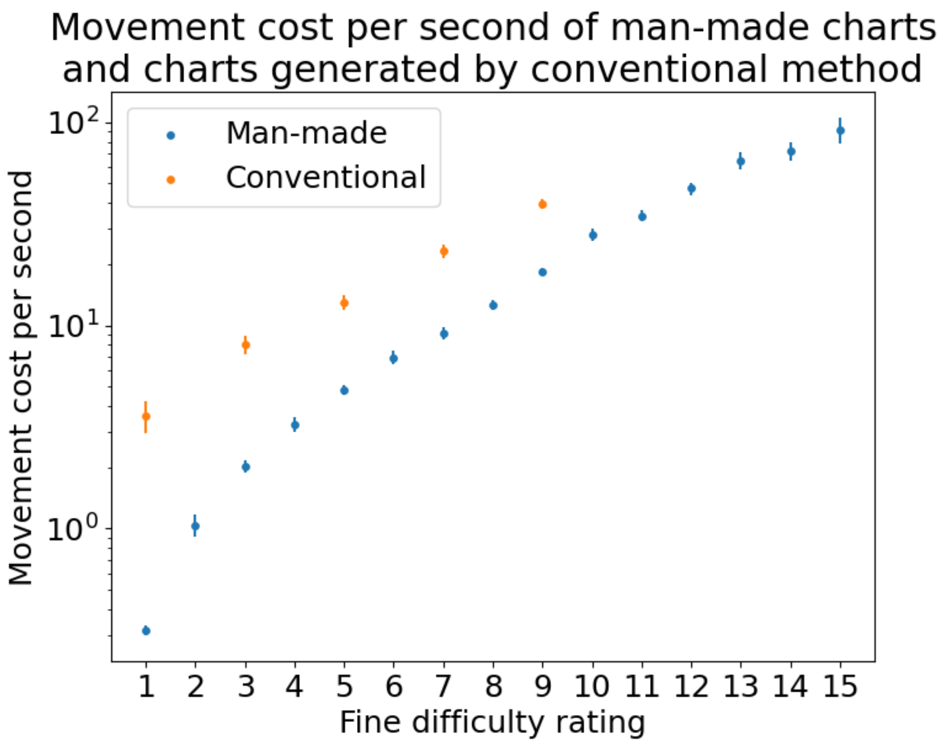

Fig. 4.7: Movement cost of charts by human and DDC, compared

By comparing the charts by movement cost, we revealed by value that the generated charts by DDC is more difficult than man-made ones. For each fine

Table 4.1: Fixed fine difficulty ratings for charts generated by DDC for each coarse difficulty rating and actual fine difficulty rated from the movement cost of generated charts.

Coarse Difficulty	Fine Difficulty rated by DDC	Fine Difficulty rated by Movement Cost
Beginner	1	4
Easy	3	7
Medium	5	8
Hard	7	9
Challenge	9	11

difficulty rating, we use the corresponding average of movement cost per second as the target difficulty for controlling the difficulty of the generated charts. In the following sections, we discuss two methods for difficulty control.

4.2 Difficulty Control by modifying target type

By the definition of movement cost, a chart has higher movement cost if the player needs to hit multiple pads in quick succession. In other words, if the chart contains a part where many arrows with different directions occur, the movement cost tends to be higher because the player will need to move the foot to one pad to another. This method modifies the direction of arrows suggested by the step selection model in the conventional method by calculating the movement cost of the generated chart.

By design, the step selection model outputs the direction of arrow one-by-one. This makes it possible to treat the output as tentative direction. With this tentative arrow direction, the movement cost of the generated chart up until that point can be calculated on-the-fly, and apply necessary modification to the suggestion when the chart being generated becomes too difficult or too easy.

4.2.1 Method overview

Figure 4.8 shows the overview of this method. For our method, the system is modified to accept a fine difficulty rating value alongside of the audio file and the coarse difficulty rating. First, the average movement cost per second for the given fine difficulty rating is multiplied by the length of the song to get the desired difficulty in terms of the movement cost. Then, the step placement model of the conventional method marks the prominent epochs in the given audio data which human charters may place arrows. The list of epochs are passed to the step selection model, where the direction of arrows are determined one-by-one. Each time the arrow direction is determined, the movement cost of the chart generated up until this point is calculated. If the calculated cost becomes greater or lesser than the desired movement cost range, our method modifies the suggested direction of the arrow to try to satisfy the movement cost requirement. If it fails to do so, the system will keep the most appropriate options possible. We set the threshold for redoing the suggestion as 15% greater or lesser than the desired movement cost.

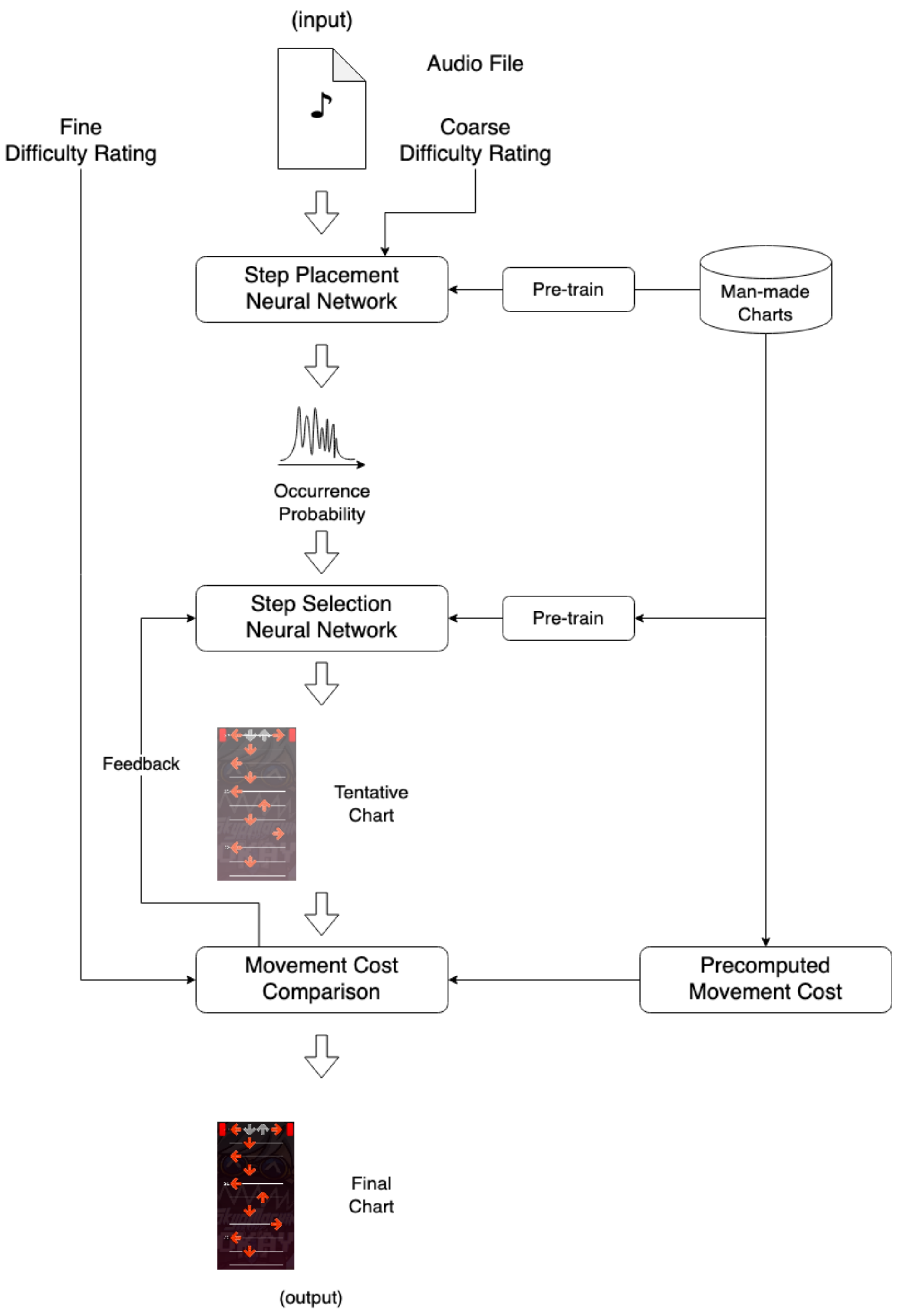

Fig. 4.8: Overview of our method to control difficulty by treating output of step selection as tentative suggestion

We aim to fine-tune the difficulty of the generated charts with this method since it mainly focuses on changing the arrow directions to reduce movement cost rather than adding or removing arrows.

The man-made charts DDC used to train its models have a specific range of fine difficulty rating for each coarse difficulty rating, which is shown in Table 4.2. Note that there was a chart with a coarse difficulty rating of "easy," but had fine difficulty rating of "12." This chart was not included in the range described in this table as it had an extremely higher density compared to other "easy" charts, and the charter presumably made this chart as a "joke chart."

Table 4.2: Range of fine difficulty ratings for each coarse difficulty rating of man-made charts

Coarse difficulty	Fine difficulty range
Beginner	1 ~ 2
Easy	2 ~ 5
Medium	4 ~ 9
Hard	6 ~ 12
Challenge	9 ~ 15

4.2.2 Experiment and result

An experiment to compare the chart generated by this method to the one generated by the conventional method was conducted. For this experiment, we changed the desired fine difficulty rating from 8 to 12 while choosing an appropriate coarse difficulty rating for each actual fine difficulty rating by consulting Table 4.1 so that the tentative charts requires only small enough modifications by our method for it to have the movement cost within the desired range. The chosen coarse difficulty rating for each fine difficulty rating is shown in Table 4.3

Table 4.3: Choice of coarse difficulty rating for each fine difficulty rating tested

Fine difficulty	Coarse difficulty choice
8	Easy
9	Medium
10	Hard
11	Challenge
12	Challenge

Figure 4.9 shows the comparison between the movement cost per second of man-made charts and that of charts generated by our method. This method appears to generate charts whose movement cost is close to man-made ones. Figure 4.10 shows some examples of a man-made chart and a chart generated by our method.

The charts were generated by specifying the same fine difficulty rating as the man-made chart as the desired rating. When compared with the man-made charts, our method seems to have generated charts that looks less complex due to the difference in the arrow density.

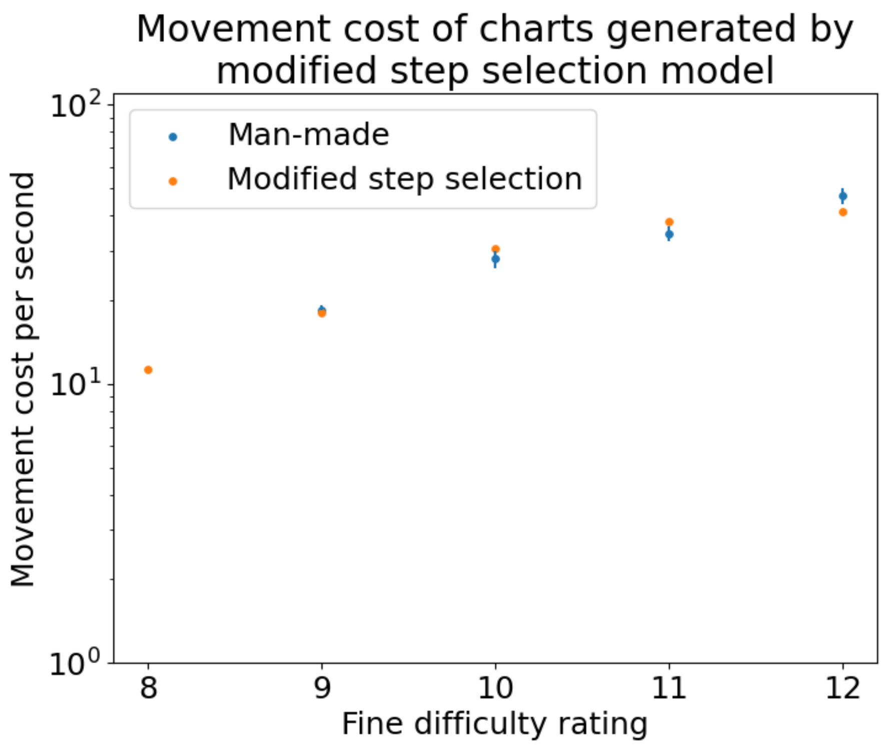

Fig. 4.9: Comparison of movement cost between man-made charts with fine difficulty rating range of 8 to 12, and charts by our method when specifying level 8 to 12 as the requested fine difficulty rating

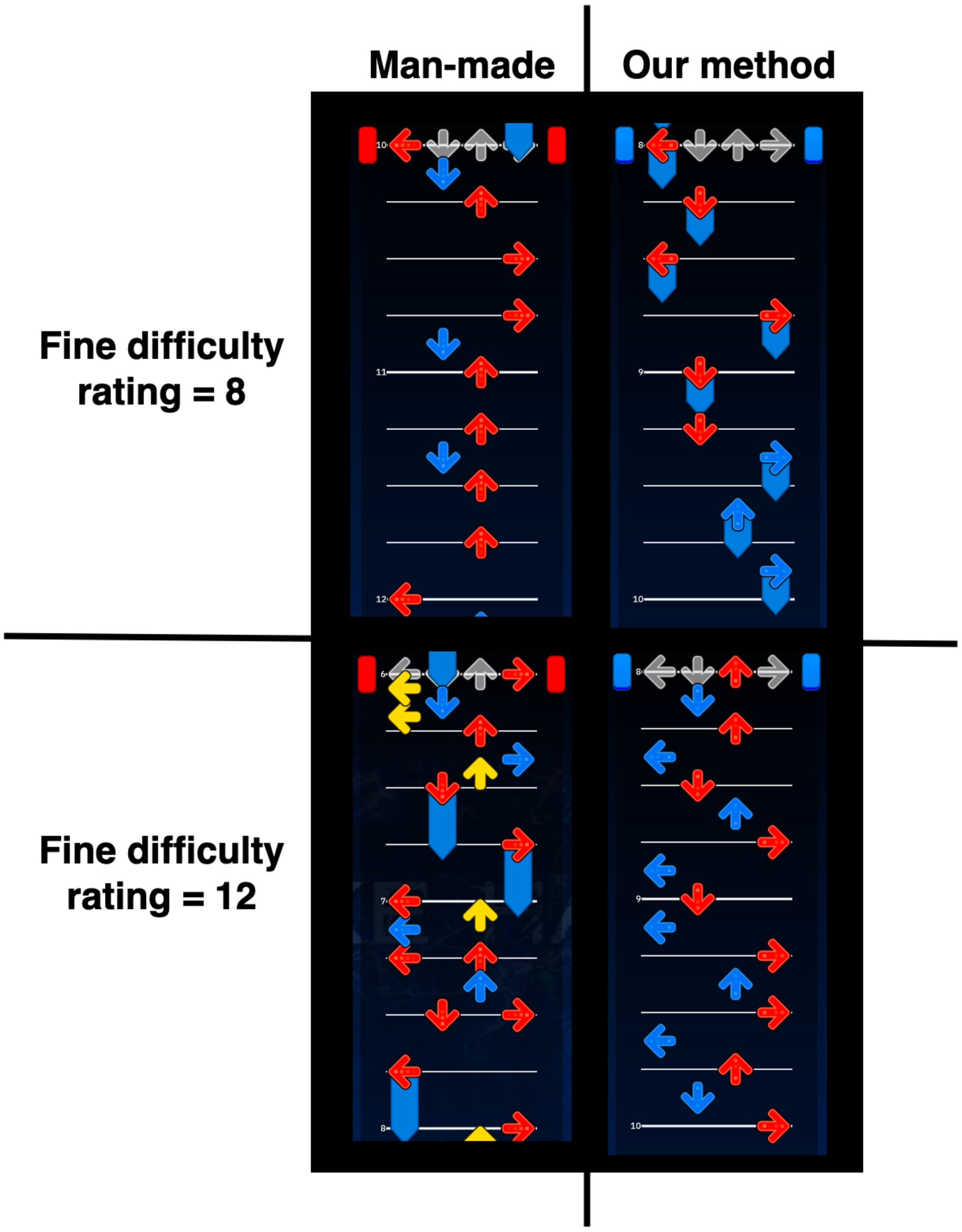

Fig. 4.10: Comparison among a man-made chart and two charts by our method, specifying the same difficulty rating as man-made ones as the request. Yellow arrows indicate that they occur at moments that are offset by either $1/4$ or $3/4$ of the temporal length between two beats from red arrows (which are on the beat).

However, in some cases, several arrows that occur in a short period of time has the same direction (Fig. 4.11), which cannot be seen in man-made charts. This unnaturalness was prominent especially when the desired fine difficulty rating was significantly lower than that of a chart without any fine difficulty rating request. For example, if the conventional method outputs a chart with the movement cost within the range of 'level 12' for a certain audio input and a requested coarse difficulty rating of 'challenge,' trying to generate a 'level 8' chart by applying our method results in multiple bursts of arrows that is in a same direction.

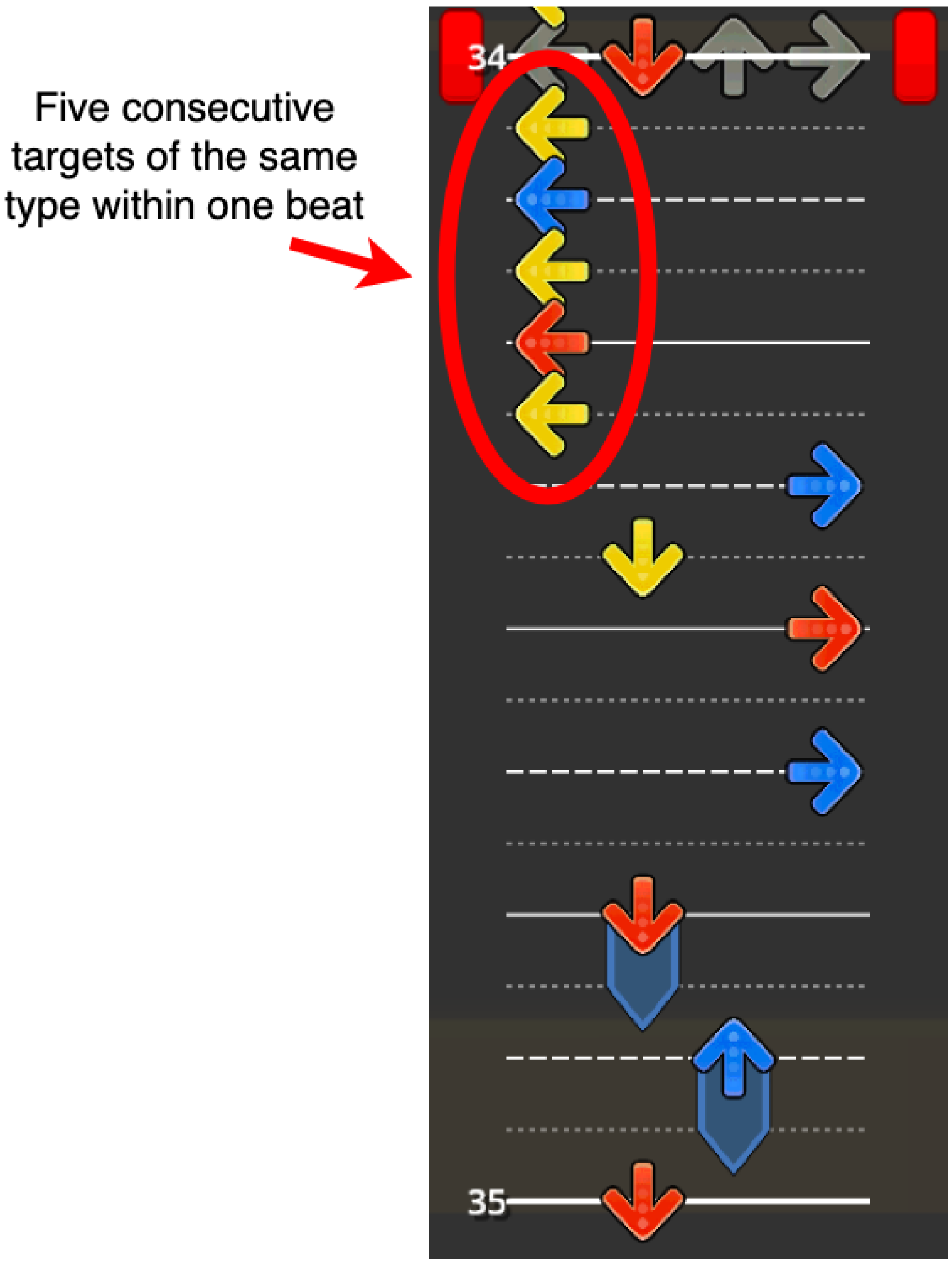

Fig. 4.11: An example of a chart generated by our method that has characteristics not seen in man-made charts

4.3 Difficulty control by modifying target count

The definition of movement cost suggests that if the number of arrows within the chart is low, then the movement cost is also low and thus the chart has low difficulty level.

An additional investigation was conducted to find the density of arrows for each fine difficulty rating. Figure 4.12 is the result where the error bar is 95% confidence interval.

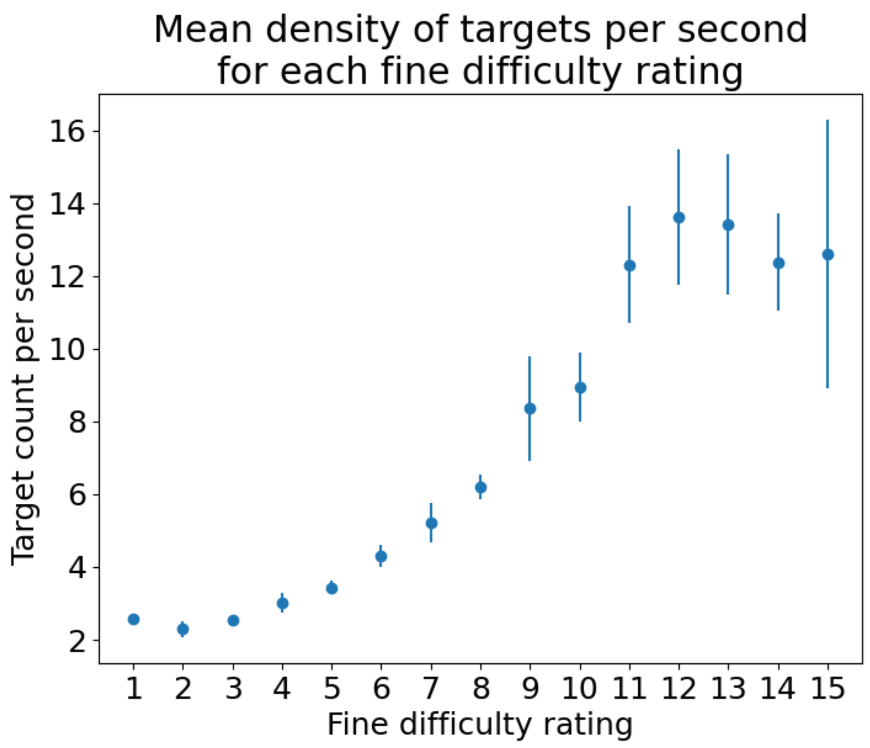

Fig. 4.12: Relation between the average arrow density of charts and each fine difficulty rating

Due to the small sample size, the interval for the fine difficulty rating of 15 is significantly large, but it was found out that the density of the arrows generally increases as the fine difficulty ratings increase. In the next section, we discuss the method to increase or decrease the number of arrows in the chart to control the difficulty of the generated chart.

The step placement model finds the prominent epochs such as onsets in the given audio signal (i.e., music) on which human charters are likely to place targets. The model accomplishes this behavior by first applying log-mel spectrum [12] to the given audio signal, which is then converted into a representation in a frequency domain called mel scale. Mel scale is weighted by the bias in human hearing, thus reflects how human listen to audio better than plain short-term fourier transform. The calculation is performed with a hop length of 512 samples, giving the spectra of audio for each slice with a length of ~ 11ms. From these samples, the model outputs a value within the range of $0 \sim 1$ which corresponds to the likelihood of a target being present for each sample. The values are then compared with a constant. If the value is greater than the said constant, the corresponding sample will be considered to contain a target. In the conventional method, the constant is defined for each coarse difficulty rating as the value that yields charts with the best F1 scores compared with the man-made chart. F1 score is a measure for evaluating the performance of a model that outputs two values (truthy and falsy). It ranges $0 \sim 1$ and will decrease if the generated data and the ground truth disagrees. If the constant is modified, more or less samples will be picked up as potential timings to place a target, which means it can control the number of targets. In our method, the constant is modified according to the required movement cost range to find the best number of arrows for the desired fine difficulty rating.

4.3.1 Method overview

Figure 4.13 shows the overview of this method. Similar to Section 4.2, this method accepts a fine difficulty rating value alongside of the audio file and the coarse difficulty rating, and the desired movement cost for the chart is the average movement cost per second for the given fine difficulty rating multiplied by the length of the song with the acceptable movement cost range being 15% greater or lesser than the desired movement one. The major difference between the previous method is that this method modifies a constant used by the conventional method to change the amount of targets. This constant is used as a cut-off value for Step Placement model's output, which is the probability of target being present at the given moment. The moments with target occurrence probability larger than the constant will have targets placed on them. In the conventional method, this threshold is predetermined from the similarity of the generated charts to the man-made ones, determined by F1 metrics. Our method changes this threshold so that the amount of the arrow changes: when decreased, the target count increases while when increased, the target count decreases. Since movement cost cannot be calculated without the direction of arrows being decided by the step selection model, the assessment of difficulty level with movement cost occurs after a chart is completely generated. If the generated charts have the movement cost outside the acceptable range, our method changes the constant within the range of 0.1 (increase if movement cost is too large, decrease if too low) and redo the generation to satisfy the movement cost requirement. If it is impossible to satisfy the requirement with the largest modification, the system aborts the adjustment and suggest other configuration. Such situation can occur if the requested coarse difficulty rating and fine difficulty rating are too far removed compared to Table 4.2 (e.g., the requested coarse difficulty rating is "beginner" but the fine difficulty is 10). We set the modification range to 0.1 as lowest the constant used by the conventional method is 0.153, and if the constant is too low, there is a risk of Step Placement model picking up noise and the system putting the target out of the rhythm in the given music. Since this method can directly change the number of arrows, it is expected that the method can change the difficulty level of the generated charts in a large amount.

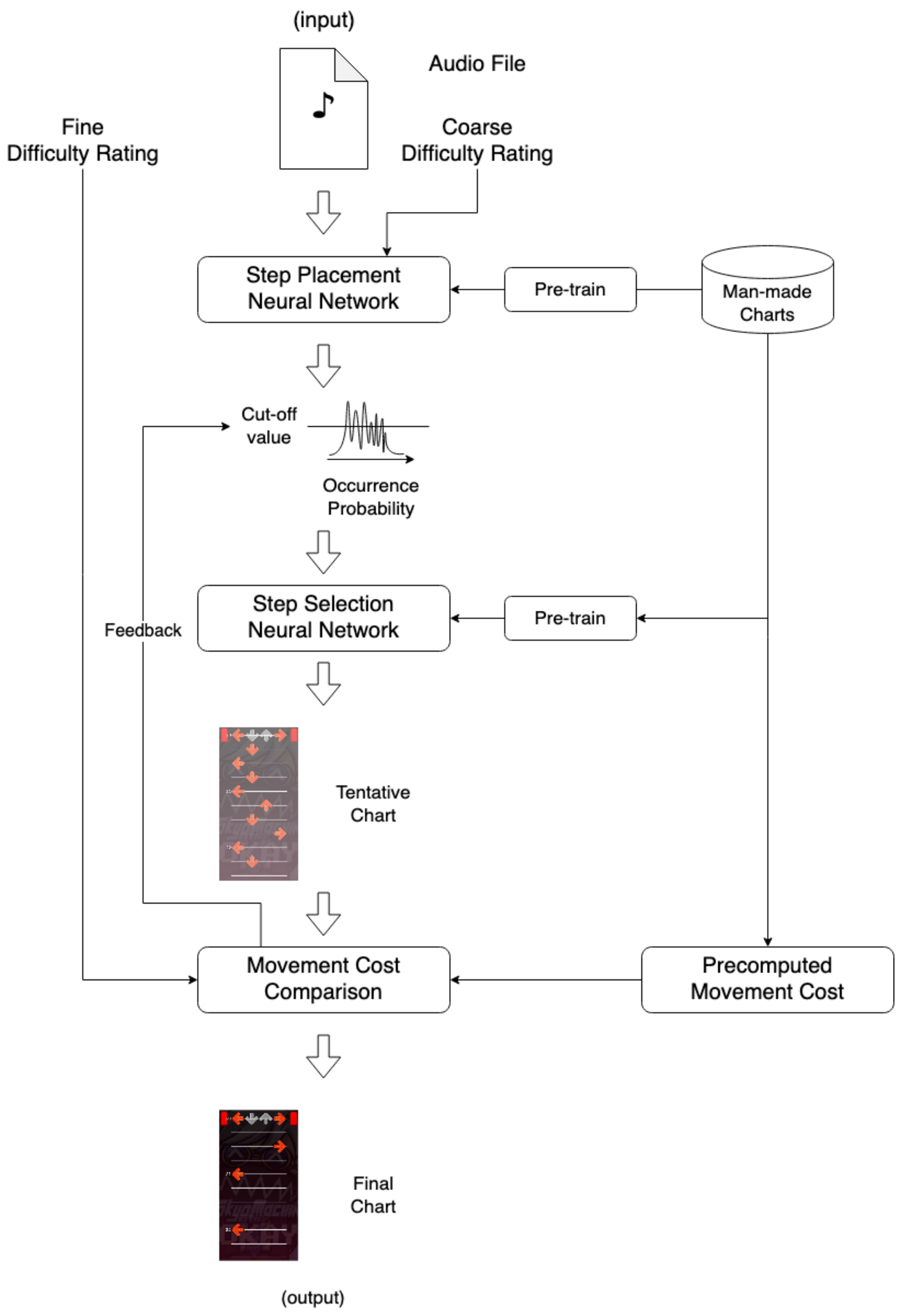

Fig. 4.13: Overview of our method to control difficulty by controlling a property of step placement

4.3.2 Experiment and result

An experiment to generate 100 charts for each fine difficulty rating was conducted. The song used is “Okay - Tokyo Machine,” which already has man-made charts but is not in the training data. The coarse difficulty rating was chosen based on Table 4.2.

Figure 4.14 shows the result. Movement cost-wise, our method was able to output charts that are close to the fine difficulty ratings requested.

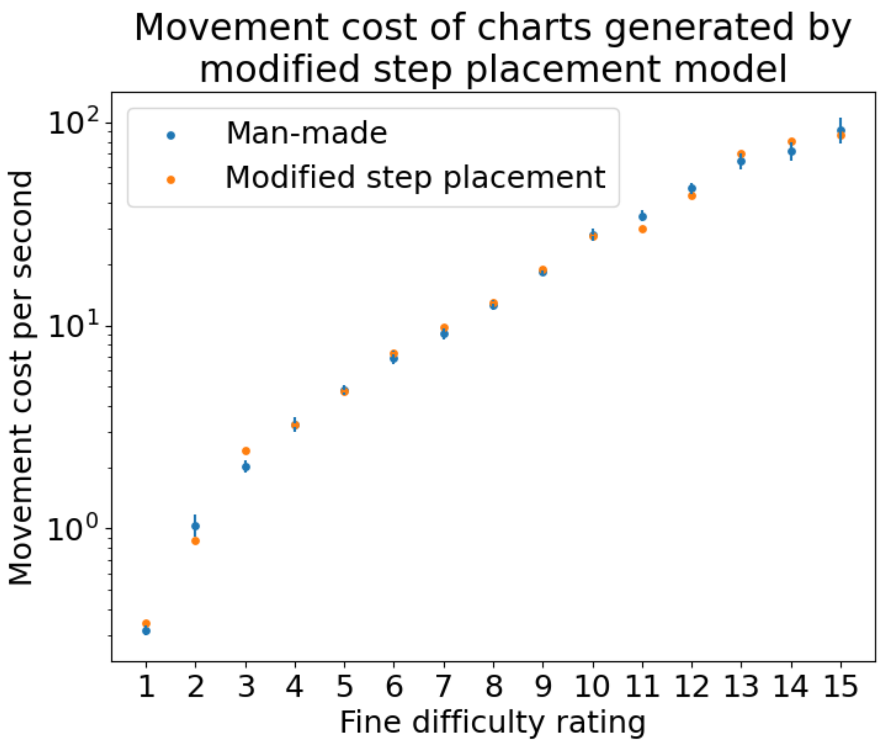

Fig. 4.14: Movement cost of charts by human and our method that modifies the threshold constant, compared

Figure 4.15 shows the examples of actual charts generated by our method compared with man-made ones. All the charts shown here have the same fine difficulty ratings of 3, and the compared excerpts correspond to the same part of the music. The man-made chart has the coarse difficulty rating of “easy.” The comparisons revealed the following charts whose characteristics cannot be observed in man-made charts:

1. Multiple sections of charts are devoid of targets. The man-made counterpart have no such extreme variation in the target density.

2. When requesting easier fine difficulty ratings than that of the tentative chart, some sections of the chart was using the off-beats, which is a timing exactly between two beats (indicated by the blue arrows in the chart). Separate research reveals that using off-beats for charts with the coarse difficulty rating of “easy” is significantly rare.

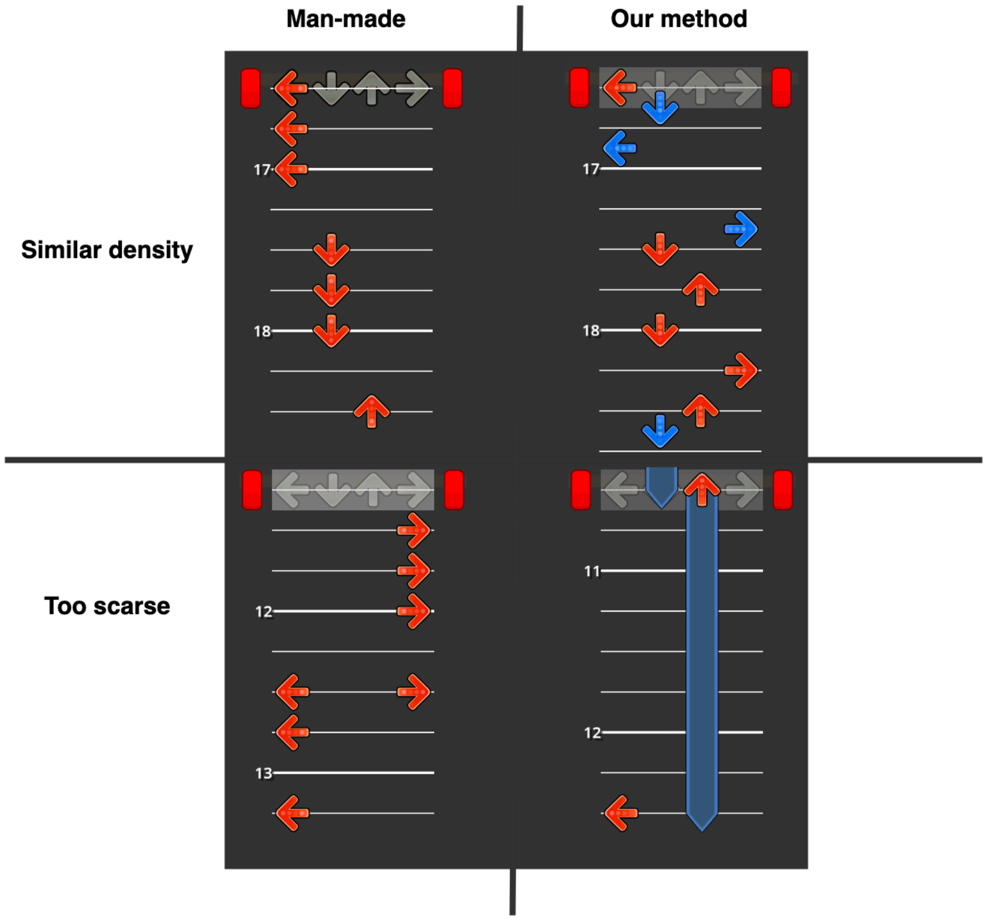

Fig. 4.15: An example of a generated chart with an extreme target density difference

4.4 Discussion

Movement cost played a role in quantifying the difficulty level of the charts by focusing on both the target density and the required movement, and both of our method showed some effectiveness in terms of increasing/reducing the movement cost to match the required difficulty level. Although taking the required movement into account may help us differentiate the difficulty levels of two charts with a similar target density, modifying the required movement to lower the difficulty level (section 4.2.1) caused the chart to become unnatural (i.e., to have characteristics not observed in man-made charts). One reason why this was the case may be that the method tried to modify the required movement in a way that the step selection model was not designed for; it was meant to receive all the candidate positions of targets from the step placement model for generating the target type, not attempting to generate the type of the same target over and over. Also, since this method does not add or remove any targets, if the tentative charts have less density than the man-made charts, our method fails to remedy such difference.

Modifying the target density by changing the cutoff value used in the step placement mode was better than modifying the movement as it could change the amount of movement needed for the player. However, this method has also generated charts whose characteristics cannot be observed in man-made charts. We suspect that this may be because the step placement model seems to have not considered the knowledge of rhythm when finding the candidate positions for the target; rather, it merely was finding the onsets within the audio. The step placement model can output more candidate positions for targets because louder parts of the music contains more audio onsets than the quieter parts. However, it does not seem to distinguish the importance of each onset, such as the type of musical instrument which caused a specific onsets to appear, or whether the onset is on the beat or off the beat. Because the model does not seem to consider human perception of rhythm such as the idea of beats and measures, which turns out to be more focused on when charters create charts with lower difficulty level, just quantifying the difficulty level and modifying the charts according to it does not introduce enough information to the methods so that the generated charts is both: 1. of a desired difficulty level, and 2. retains the characteristics of man-made charts.

In the next chapter, we propose a method that put emphasis on rhythm by classifying targets based on a certain representation of rhythm.

Chapter 5

Difficulty Control by Data from Musical Point of View

In the previous chapter, we found that the conventional method has difficulty in generating charts with appropriate difficulty level when the user requests easier coarse difficulty ratings such as “beginner” and “easy” (Table 4.1). In this chapter, we discuss a method to improve the ability to control the level of difficulty for such easier coarse difficulty ratings by consulting statistics of man-made charts from the musical point of view. Specifically, we discuss the rhythm representation for classifying targets within a chart.

5.1 Rhythm representation

The editor function of StepMania allows charters to create their original charts. It offers quantized rhythm representations for charters to accurately place the targets. The representation involves subdividing a length of time called a ‘measure’ or ‘bar,’ which corresponds to four times the length of a 4th note (which is equal to length between two beats). StepMania equally subdivides a measure in different ways to represent timings of targets in a chart in a quantized manner. The onset of each subdivision is a candidate position at which the human charter can place a target. The subdivisions and the onsets are represented by the length of a subdivision in units of note values, i.e., 4th note, 8th note, 12th note, etc., where 4th note means the onset of subdivisions of a measure equally divided into four rather than the note value itself. Since the length of 4th note can change if a song employs different tempo, or beat per minute (BPM), this representation requires the knowledge of BPM.

Using this concept, DDC-generated targets can be classified using the representation of candidate positions within a measure. There are two possible ways to represent such positions for our target classification task, as described below.

5.1.1 Index as rhythm representation

The smallest subdivision that StepMania can process is 192nd, or $1/192$ of a measure. We can assign indices from 0 through 191 for each onset of the subdivision to classify the targets on it. For example, a target at the beginning of a measure is index 0, and a target at the first off-beat within a measure is index 24. We call this method of classification "positional classification," and each class "positional class."

5.1.2 Note value name as rhythm representation

For the representation of rhythm by note value names, we use all the representation StepMania offers the charters, i.e., 4th, 8th, 12th, 16th, 24th, 32th, 48th, 64th, 96th, 128th and 192th notes, plus the longer 1st and 2nd notes to cover intervals between targets longer than that of two beats. Let us explain the details of the classification with a target that can be classified as the 4th note as an example. This target can appear at one of the four subdivision onsets. Correspondingly, it can be classified as 4th. However, multiple candidates of the classes can exist for a single target. For example, a target appearing at the very beginning of a measure can be classified as any of the 13 classes. To avoid such ambiguity, we choose the class with the roughest subdivision among the possible candidates for that target. Figure 5.1 shows an example of the rhythmic positions and the target classification (bottom). It is important to note that there are finite numbers of possible moments in a measure to be classified as those classes.

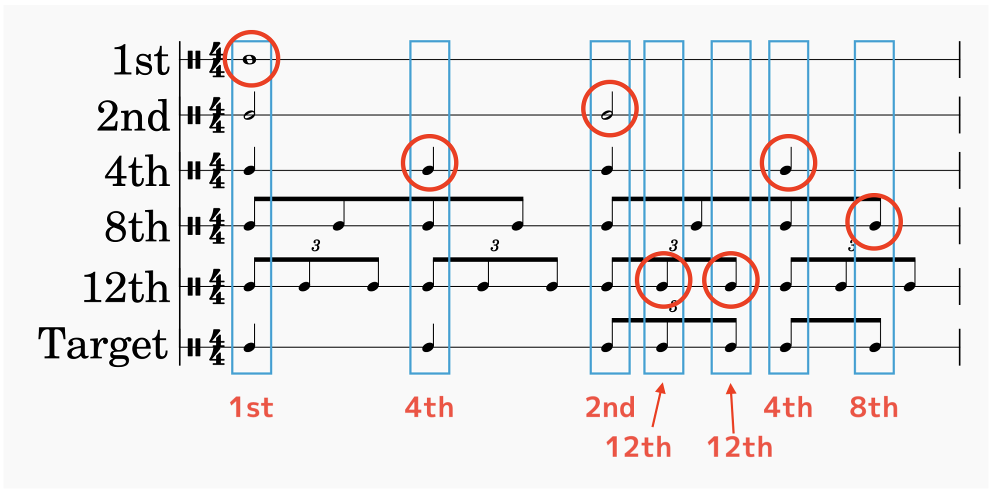

Fig. 5.1: Classification of an example target rhythm using note value to define note position.
Note that on multiple matches, the smallest subdivision is used (circled in red).

Figure 5.2 and Table 5.1 show the maximum possible numbers of targets per measure. The maximum possible number of targets per measure for each class is important because it can be used to calculate the occurrence frequency. We refer to this method of classification as "subdivision classification" and the classes as "subdivision classes."

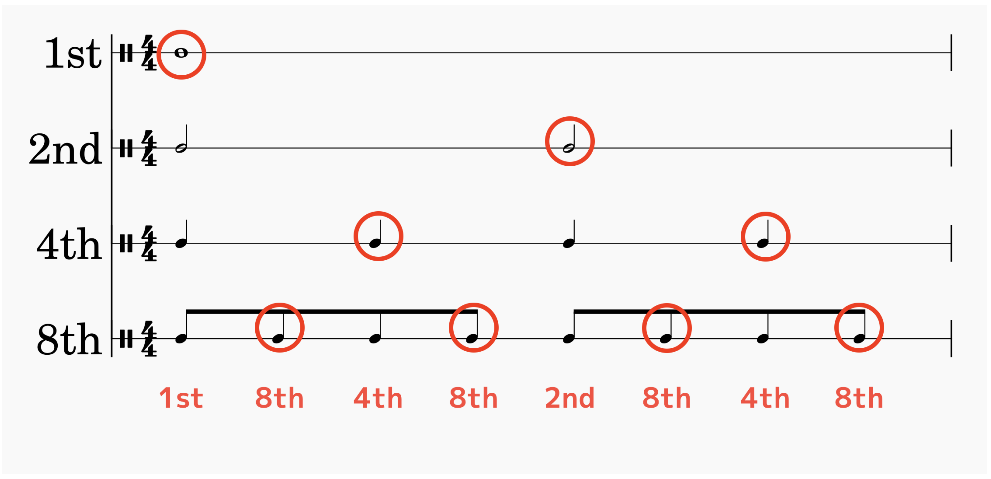

Fig. 5.2: Illustration of the maximum number of moments for each subdivision class

Table 5.1: Maximum possible count of TPM.

Class	Maximum count
Whole Note	1
Half Note	1
4th Note	2
8th Note	4
12th Note	8
16th Note	8

5.1.3 Target per measure

Since the charts vary in length, the target count for each class should be normalized. We choose the temporal length of a measure for normalization because the classes are dependent on the measures. We refer to the normalize target counts as "target per measure," or TPM.

After the classification of all the targets in the chart, the TPM for each class is computed by dividing the number of targets in the same class by the total number of measures. The reference TPM is computed as an average of TPMs for all the man-made charts. That is,

$$\bar{T}_{c,l} = \frac{1}{N_l} \sum_{k=1}^{N_l} \frac{T_{k,c,l}}{m_k}, \tag{5.1}$$

where c and l indicate one of the positional/subdivision classes and the difficulty level, respectively. N_l is the number of man-made charts with difficulty level l, $T_{k,c,l}$ is the number of targets in class c for chart k with difficulty level l, and m_k is the number of measures of the song corresponding to chart k.

5.2 Target Occurrence Frequency for Each Rhythm Representation

An experiment was conducted to calculate the target occurrence frequency for each class for both rhythm representations. The man-made charts used for the experiment is the dataset used to train the models of the conventional method, namely, 450 charts created by the same charter which contains a total of 350,000 targets.

Figure 5.3 shows the occurrence frequency for each positional class. For easier coarse difficulty ratings such as ‘beginner,’ the number of peaks are smaller, meaning that the charter focuses on beats rather than the onsets of the sound within the song. For more difficult ones such as ‘challenge’, more peaks can be observed, which means the charter puts more focus on the onsets of the sound within the song.

Note that the occurrence frequency for some classes is zero or significantly close to zero. We find that the classes with an index that is a multiple of 12 have noticeable occurrence probabilities, but not the ones with an index of a multiple of 4, 6, or 8.

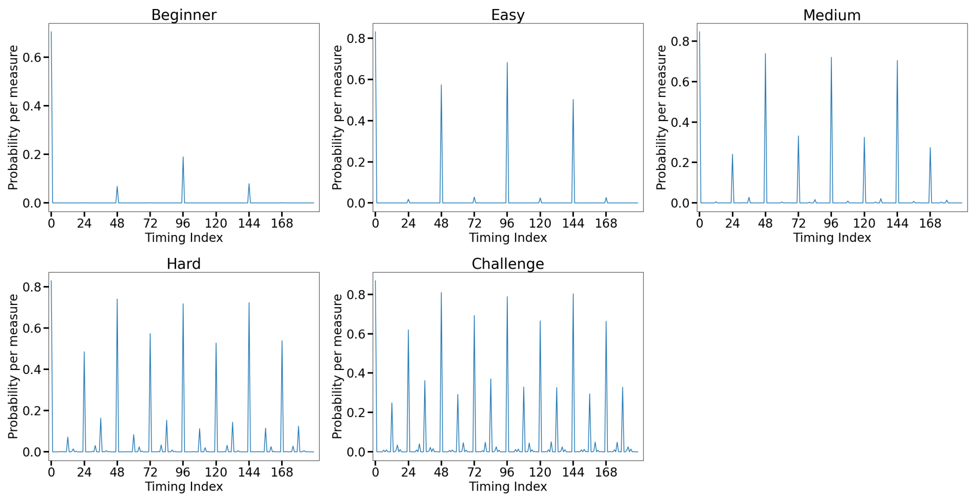

Fig. 5.3: Probability of targets for each positional class per measure and for each coarse difficulty in man-made stage data. Ticks on X-axis represent 8th timings.

Figure 5.4 shows the occurrence frequency for each subdivision class. Since all the subdivision classes finer than 48th had the occurrence frequency of 0 or close to 0, it is removed from the figure. Similar to the positional classification, the higher the coarse difficulty rating is, the more targets classified into finer subdivision classes can be observed.

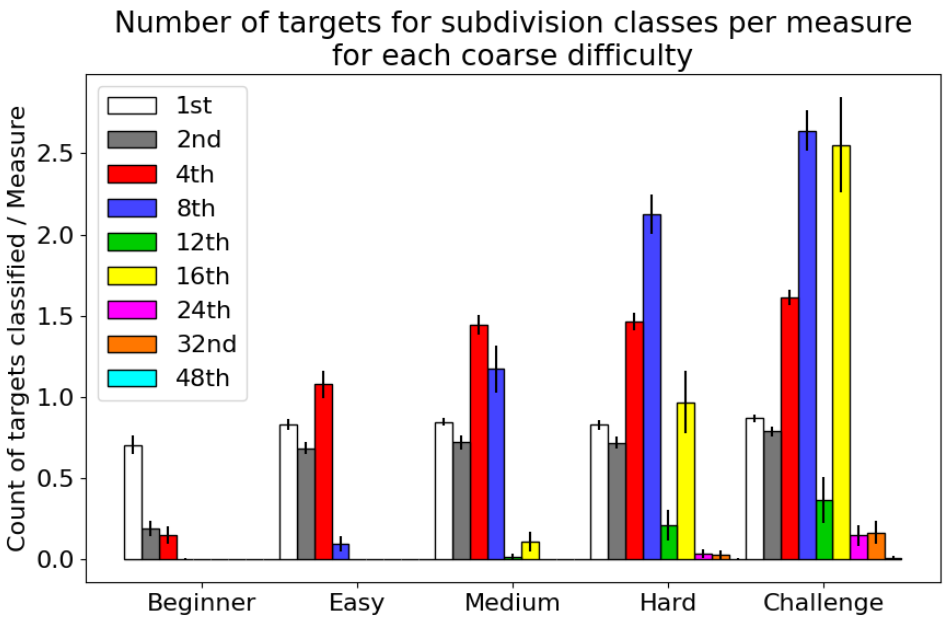

Fig. 5.4: Average number of targets per a measure for each subdivision class and for each coarse difficulty in stage data.

For both rhythm representation, there were clear differences in the target occurrence frequency among the coarse difficulties. This means we can use this data as a reference to control the difficulty level of the generated charts; the targets in any generated chart can be classified using the technique mentioned above, and remove the targets from the said chart if the target occurrence frequency for any given class is too high. Also, it was found that we do not necessarily have to consider all subdivisions that StepMania supports since there are no, or almost no targets classified as a class with extremely fine of subdivision (e.g., index 1 for positional classification, and the 96th class for subdivision classification). For this reason, we can disregard such classes to reduce computational cost and potential noise. When disregarding classes, the remaining classes must satisfy the following criteria: 1. Classes with high occurrence probability are properly represented. 2. Triplets, which are 3 subdivisions of equal length of a given subdivision, can be represented. With these restrictions in mind, for positional classes, we chose the subdivision count of 48 instead of 192 as explained in subsection 5.1.1 and indexed the classes accordingly to use them in our method. For subdivision classes, we simply disregarded subdivision classes finer than 48th. In both cases, our TPM can express the possible positions of all common notes while being able to handle triplet cases up to 24th notes.

In the next section, we will discuss how to integrate reference TPMs into the difficulty control method.

5.3 Method Overview

Figure 5.5 shows an overview of our system. Our system takes a musical audio clip and a coarse difficulty rating as input and generates a chart. We assume that

BPM of the input music is also provided. The system consists of a deep neural network and a refinement filter. The audio clip and the difficulty level are first fed into the network to produce a tentative chart. The network is pretrained to detect the prominent onsets of the input music and the targets are placed at the detected moments.

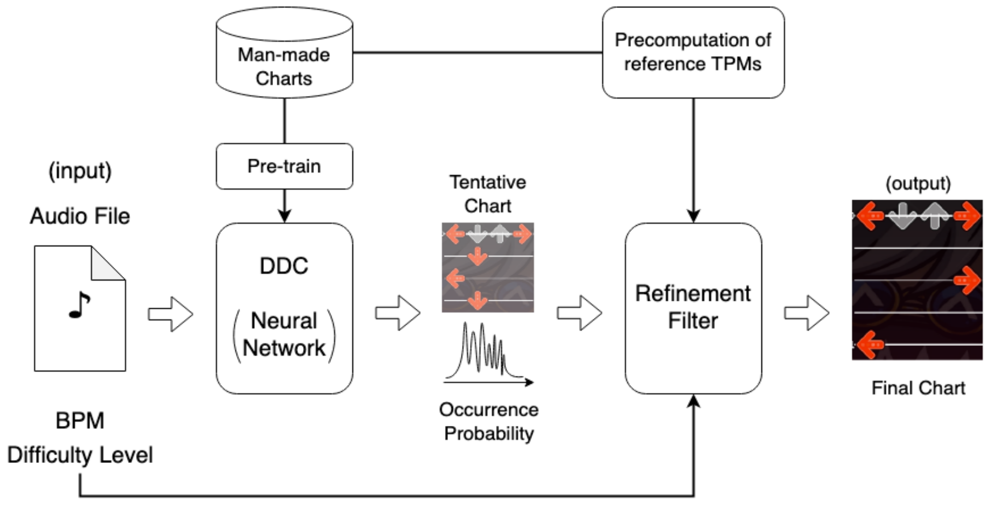

Fig. 5.5: Overview of our method.

The tentative chart is often too difficult, that is, the number of targets is larger than expected. Our refinement filter addresses the problem by removing some targets to match the specified difficulty level. TPMs we calculated in the previous section is used for this task. The filter removes targets in the tentative chart so that the average TPM becomes the same as the reference TPM for the specified coarse difficulty rating.

5.3.1 Refinement filter

Our method first classifies the targets in the tentative chart into several classes in accordance with their corresponding rhythmic positions, and TPM for each of the classes is computed. The TPMs to be used are shown in Figs. 5.3 and 5.4.

First, each target in the tentative chart is classified into one of the classes in accordance with their rhythmic positions, as described in the previous sections. The chart is then subdivided into a set of short temporal segments; we use eight-beat (or two-measure) segments in our method. The refinement process is applied to each of the segments sequentially. The preferred number of targets for class c in a segment, P_c, is calculated using the reference TPM, that is,

$$P_c = 2 \times M_c \times \bar{T}_{c,l}, \tag{5.2}$$

where M_c is the maximum possible count of targets within two measures for each class ($M_c = 2$ for all positional classes, and see Table 5.2 for M_c of each subdivision class) and $\bar{T}_{c,l}$ is the reference TPM for class c with difficulty l. P_c can be a fractional number. Thus, the integer part of P_c is considered to be the number of targets to be always included in the segment while the fractional part is used as a probability of whether one extra target is preserved.

Table 5.2: M_c for each subdivision class

Subdivision class	M_c
Whole Note	2
Half Note	2
4th Note	4
8th Note	8
12th Note	16
16th Note	16

For the removal part, we use the target occurrence probability computed by DDC for each target. The targets are first grouped by their subdivision classes, and then they are sorted by the occurrence probability within the group. The target with the lowest probability is removed first until the desired number for the subdivision class is satisfied.

Note that our refinement method does not add any targets because it lacks the ability to analyze the input audio signal for appropriate target placement. If the number of targets for the given subdivision class is already smaller than the desired target count, the removal process is skipped.

5.4 Result

5.4.1 Pilot study

A preliminary experiment to compare the performances of the methods using positional class and subdivision class was conducted. For this experiment, we used the reference TPM for beginner charts and 10 songs that have man-made charts, which we use as ground truths. To train the model in the conventional method, 91 songs annotated with stage data by the charter Fraxtil were used [13]. The total length of the songs is more than 200 minutes and all of the songs have stage data for all 5 possible coarse difficulties. The same set of stage data is used in our method to calculate the reference TPMs.

Figure 5.6 shows the mean arrow density of the chart for each method. The subdivision class method generates charts with target densities closer to those of the man-made charts.

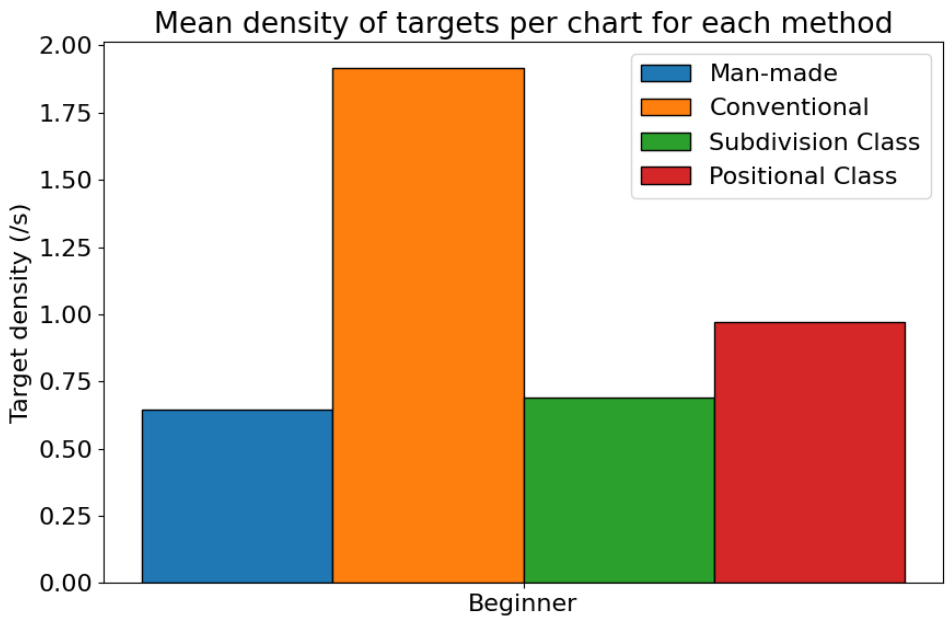

Fig. 5.6: Mean arrow density among the man-made chart, the conventional method, our method using subdivision class, and our method using positional class. Sample size: 10 songs.

Table 5.3 lists the F1 score for each method relative to the corresponding man-made charts. Comparing the methods by the F1 scores reveals similar results; the subdivision class method exhibits slightly better performance. Note that the F1 score shown in this table and hereinafter is the average of each attempt at generating a chart.

Table 5.3: Mean precision metrics of the conventional method, our method using subdivision class, and our method using positional class, for the beginner difficulty level.

Method	Precision	Recall	F1
Conventional	0.264	**0.658**	0.363
Subdivision Class	**0.413**	0.501	**0.463**
Positional Class	0.371	0.434	0.400

On the basis of the results, we decided to further compare the subdivision class method with the conventional method to expand Donahue's[1] research to the occurrence of targets counted by a similar concept.

5.4.2 Comparison between the conventional method

An experiment to compare the performances of the conventional model [1] and our method was conducted.

Since it is known that the difficulty is proportional to the density of the targets in a chart, an experiment to compare the density of arrows in charts made by humans, by the conventional method, and by our method was conducted first. Figure 5.7 shows the result of the experiment. In addition to the increase in the density in accordance with the coarse difficulty for the man-made chart, our method generates charts that have densities closer to those of the man-made charts than does the conventional method for difficulties of the medium or easier levels.

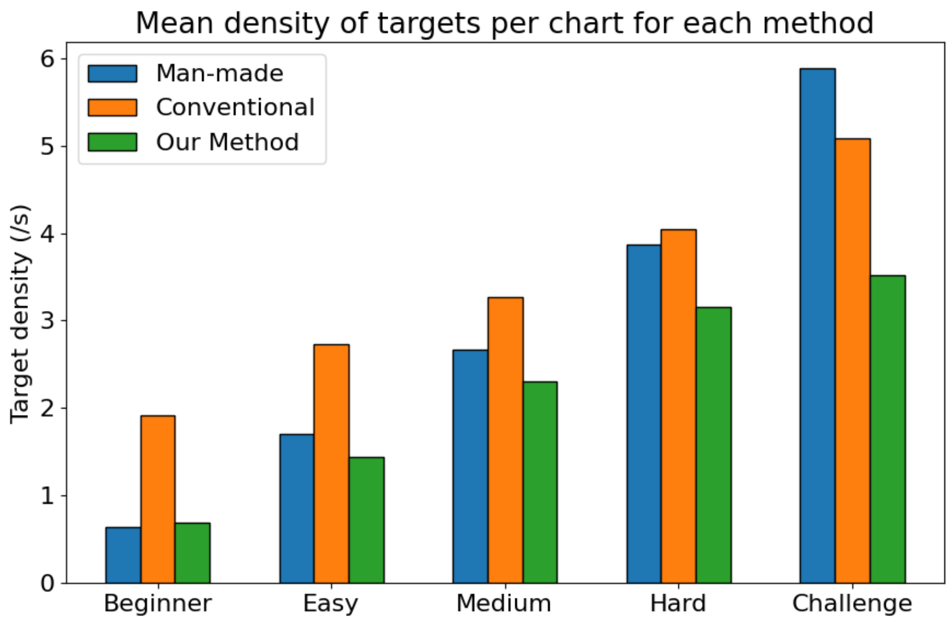

Fig. 5.7: Mean arrow density of the man-made chart, the conventional method, and our method. Sample size: 10 songs for each difficulty.

The charts generated in this experiment were then analyzed using the F1 score metrics to evaluate how close they are to man-made charts. Table 5.4 shows the average precision, recall, and F1 score of the conventional method and the proposed method for generating 10 charts for each difficulty level. The comparison is done by calculating the F1 score of the generated charts and comparing them with the corresponding man-made charts for the same song. On calculating the score, a target in the generated charts is considered as true positive if there is a corresponding target in the man-made chart within a 23 ms temporal difference as per Donahue's[1] evaluation method. For lower levels of difficulty, our method has better precision than the conventional method while the conventional method performed better in recall. As the level of difficulty increases, the difference between our method and the conventional method becomes small until our method no longer performs better than the conventional method.

Table 5.4: Mean precision metrics of the conventional method and our method for each difficulty level.

Level & Method	Precision	Recall	F1 Score
Beginner, Conv.	0.264	**0.658**	0.363
Beginner, Proposed	**0.413**	0.501	**0.463**
Easy, Conv.	0.479	**0.864**	**0.611**
Easy, Proposed	**0.633**	0.501	0.51
Medium, Conv.	0.657	**0.937**	**0.772**
Medium, Proposed	**0.693**	0.570	0.61
Hard, Conv.	0.786	**0.927**	**0.850**
Hard, Proposed	**0.793**	0.550	0.64
Challenge, Conv.	**0.857**	**0.922**	**0.889**
Challenge, Proposed	0.76	0.61	0.673

Figure 5.8 shows the counts of targets with each subdivision class for the conventional method and our method, as well as the reference TPMs that our method is aimed at (from Figure 5.4). With our method, the ratio between the classes is close to that of the reference TPMs, indicating that our method succeeds in

rectifying the chart from the conventional method to match the reference.

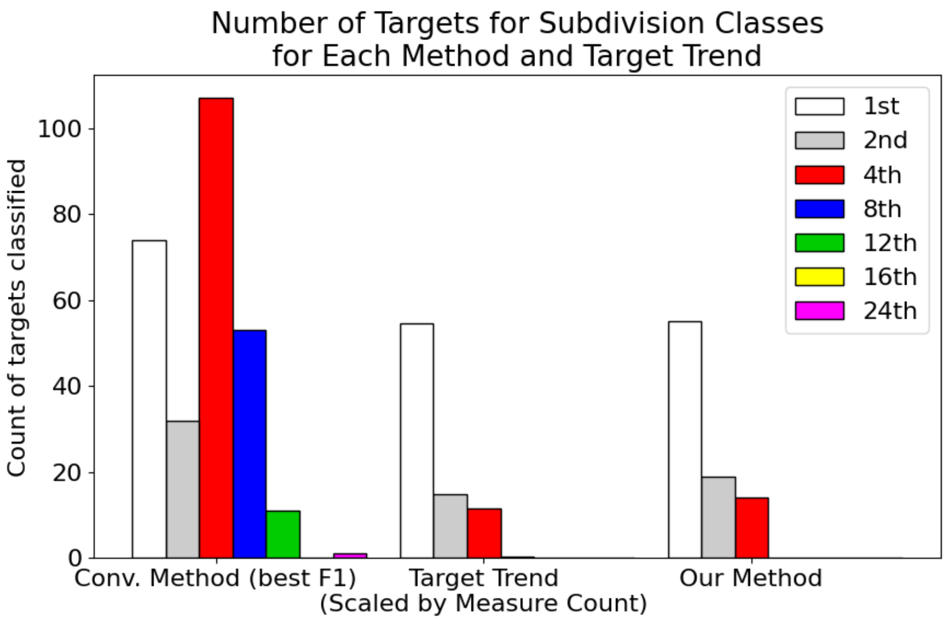

Fig. 5.8: Target counts for one of the suggestions by the conventional step placement, the target trend of our method, and the result of our method.

A visual comparison of the methods was conducted by creating playable charts from the suggestions generated by the conventional method and our method. Figure 5.9 shows excerpts of the resulting charts plus the man-made chart for reference. Note that only the number and the timing of the targets are computed but the direction of the arrows associated with the targets is not. The directions of each target in Fig. 5.9 are chosen by hand by the author since our method only suggests the temporal positions of the arrows. It is clear that the number of targets suggested by our method is closer to the man-made data, and fewer false positives occur than by the conventional method.

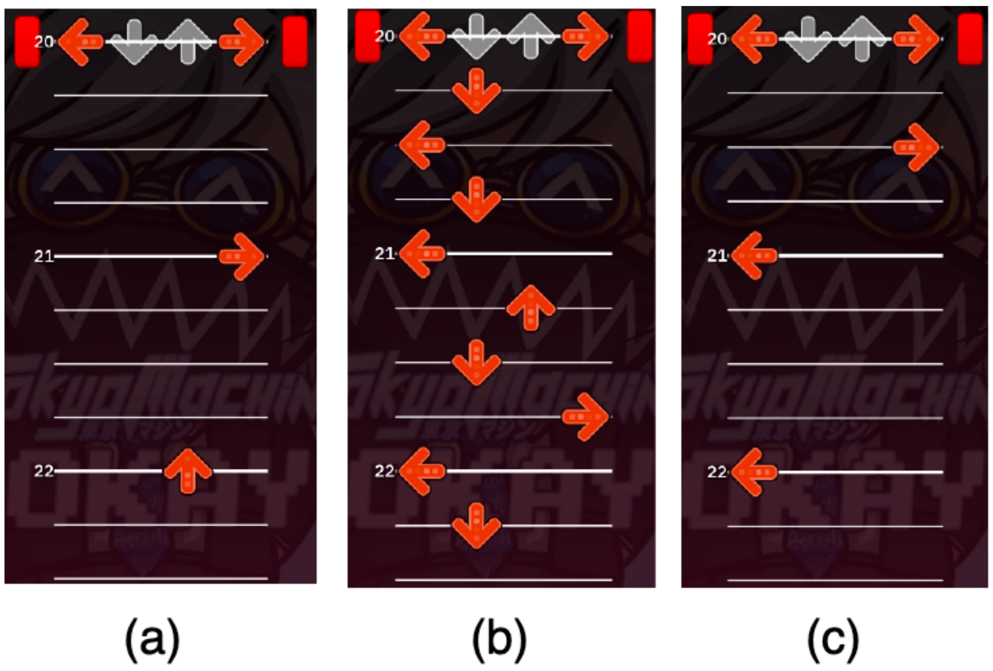

Fig. 5.9: Examples of target placement by (a) the man-made chart, (b) the chart using moments suggested by conventional method, and (c) the chart using moments from our method for the music Okay - Tokyo Machine.

5.5 Discussion

We have demonstrated that our method of using a filter that employs rhythm representation in conjunction with the conventional model yields better difficulty control when generating sparse stage data with lower difficulty. Our refinement filter adds an extra layer of knowledge that the conventional model has trouble inferring, which may have prevented it from generating a set of target candidates with the desired difficulty.

The limitations of our current implementation include the following.

1. It is not effective for stage data with higher levels of difficulty.
2. The user must input BPM and the offset of the first beat, which was not required in the conventional method.
3. The filter will produce an incorrect output if the music given has varying BPM.
4. Dependence of difficulty on BPM are not considered.
5. Some music may not employ 4/4 measure, but the filter assumes that it does.
6. Since the subdivision class trend is the average across the stage data used to train the model, our method may produce results that are too similar results among different audio data inputs.

The first limitation may come from two reasons: 1. Our method can only remove the targets when targets of specific classes are too abundant, but does not add any if they are too sparse. 2. As the level of difficulty increases, the underlying conventional method does not generate charts dense enough compared with the man-made one (see Fig. 5.7), making any attempt by our method ineffective. Developing a reliable way to add targets may solve this limitation.

The second and third limitations can be resolved by using some existing methods for the automatic detection the BPM of the given music.

It is worth noting that BPM affects the temporal length of measures, which will then affect the frequency required for the player to hit the targets. While music with excessively high or low BPM can change the difficulty of the chart for the same TPM (hence the fourth limitation) we excluded this issue from our scope since such differences in difficulty should not be expressed by the coarse difficulty labels (e.g., beginner, easy and medium) which are difficulties of different charts for the same music, but by fine difficulty labels (i.e., an integer value) which indicate the difficulty of the chart throughout the entire game. Employing fine difficulty labels may further improve the appropriateness of the generated charts.

It is another challenge to solve the fifth limitation, since most of the music used in music games employ 4/4 measure, leading to a scarcity of datasets of non-4/4 measure music compared with 4/4 measure music. The perceptual meaning of subdivision can change if non-4/4 measure is employed, e.g., half notes in 3/4 measure music are perceived differently using the same subdivision as in 4/4 measure music. For this reason, the refinement filter may need to weigh the subdivision before performing filter operations to support the target moment generation task for such music.

The last limitation was discovered when some cases were found where the F1 score for the individual result of our method was worse than the conventional counterpart. Investigations on such cases revealed that the corresponding man-made stage data had different trends than what Fig. 5.4 shows. This result was due to the lower recall of the stage data generated by our method; if the corresponding man-made stage data employs a different trend, our method may fail to place the target at the position in the man-made chart (false negative). While our method can generate charts with better control of difficulty than the conventional method, there is room to improve the performance in capturing the nuance inferred from man-made stage data. The implementation of such improvements is left for our future work.

Chapter 6

Conclusion

In this paper, we covered two methods of controlling the difficulty level of the machine-generated stage data for music games: 1. Quantifying the difficulty itself by simulating an optimized movement required by the chart, and modifying the generated charts from two aspects: target count and target type, and 2. Classifying the targets in the man-made charts from the musical point of view, and modifying the number of targets in the generated charts. Both approaches involve creating a reference of the difficulty level of man-made charts, considering the machine-made charts as tentative, and fine-tuning them for the player's desired difficulty level. Introducing the new knowledge, (i.e., rhythm), to the conventional method, seems to have had the best effect in improving the difficulty control ability while retaining the ability to generate charts that are similar to man-made ones. This may be because that knowledge was the aspect which the charters focus on when creating charts, but the models in the conventional method had trouble inferring such a concept. While it is difficult to measure the appropriateness of the data generated by a machine-learning approach for reasons such as the bias of training data differing from real-life ones, combining a statistical approach with the machine-learning one makes it possible to introduce a measurement of appropriateness that can also be used for postprocessing of the generated data. We consider that this combined approach should be considered for other techniques that utilize only the machine-learning approach.

Acknowledgements

This research was conducted at Information Media Environment Lab, Graduate School of Information Science and Technology of Hokkaido University. I'd like to express my gratitude for my professors, Dr. Yoshinori Dobashi, Dr. Naofumi Aoki, and Dr. Tsuyoshi Yamamoto, and to my colleagues of this laboratory for their kind and thoughtful inputs towards my research. Additionally, this endeavor would not have been possible without the generous support from JST SPRING, who financed my research.

Lastly, I would like to mention my family, especially my parents, Yuji Udo and Junko Udo, and my sister, Rina Udo. Their belief in me has kept my spirits and motivation high during this process.

This work was supported by JST SPRING, Grant Number JPMJSP2119.

Appendix A

Appendix

A.1 Author’s publications

A.1.1 Journal papers (with review)

- Atsuhito Udo, Naofumi Aoki, Yoshinori Dobashi, “Automatic Generation of Stage Data for Music Games with Sparse Target Density,” Acoustical Science and Technology, **44**, In printing.

A.1.2 International conference (with review)

- Atsuhito Udo, Naofumi Aoki, Yoshinori Dobashi, Tsuyoshi Yamamoto, “Evaluation of Machine-Made Scores for Music Games Generated by a Deep Neural Network,” IEEE The 1st International Conference on Artificial Intelligence in Information and Communication 2019 (ICAIIC2019), pp. 128-131 (2019/2)

- Atsuhito Udo, Naofumi Aoki, Yoshinori Dobashi, "Examination of Improving Machine-Made Charts for Music Game Generated by a Deep Neural Network," International Workshop on Smart Info-Media Systems in Asia (SISA2020), pp. 30-34 (2020/12)

A.1.3 Domestic conference

- 有働 篤人, 青木 直史, 土橋 宜典, 山本 強, "リズムゲームの譜面データの機械学習による自動生成," 電子情報通信学会技術研究報告応用音響研究会資料, EA2018-27, July 24–25, 2018.
- 有働 篤人, 青木 直史, 土橋 宜典, 山本 強, "リズムゲームの譜面データの機械学習による自動生成," 電気・情報関係学会北海道支部連合大会, Oct. 2018.
- 有働 篤人, 青木 直史, 土橋 宜典, 山本 強, "リズムゲームの譜面データの機械学習による自動生成手法の評価" 日本音響学会春季研究発表会, March, 2019.
- 有働 篤人, 青木 直史, 土橋 宜典, "リズムゲームの譜面データの機械学習による自動生成," 電子情報通信学会技術研究報告応用音響研究会資

料, EA2019-2, July 16–17, 2019.

- 有働 篤人, 青木 直史, 土橋 宜典, “機械学習を用いたリズムゲームのステージデータの自動生成の検証,” 電子情報通信学会技術研究報告応用音声研究会資料, SP2021-10, June 18–19, 2021. (音学シンポジウム 2021, 第 131 回 音楽情報科学研究会, 第 137 回 音声言語情報処理研究会 共催研究会, 電子情報通信学会・日本音響学会 音声研究会 連催研究会)

- 有働 篤人, 青木 直史, 土橋 宜典, “機械学習によるリズムゲームのステージデータの自動生成における難易度調整に関する一考察,” 電気・情報関係学会北海道支部連合大会, Nov. 2022.

References

[1] C. Donahue, Z. C. Lipton, and J. McAuley, “Dance dance convolution,” in Proceedings of the 34th International Conference on Machine Learning, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 06–11 Aug 2017, pp. 1039–1048.

[2] J. Hoysniemi, “International survey on the dance dance revolution game,” Comput. Entertain., vol. 4, no. 2, pp. 8–es, apr 2006.

[3] R. Zhou, M. Mattavelli, and G. Zoia, “Music onset detection based on resonator time frequency image,” IEEE Transactions on Audio, Speech, and Language Processing, vol. 16, no. 8, pp. 1685–1695, 2008.

[4] R. Zhou and M. Mattavelli, “A new time-frequency representation for music signal analysis: Resonator time-frequency image,” in 2007 9th International Symposium on Signal Processing and Its Applications, 2007, pp. 1–4.

[5] J. Schlüter and S. Böck, “Improved musical onset detection with con-

volutional neural networks," in 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014, pp. 6979–6983.

[6] Y. Liang, W. Li, and K. Ikeda, Procedural Content Generation of Rhythm Games Using Deep Learning Methods, 11 2019, pp. 134–145.

[7] E. Halina and M. Guzdial, "Taikonation: Patterning-focused chart generation for rhythm action games," 08 2021, pp. 1–10.

[8] R. Bååth, "Subjective rhythmization: A replication and an assessment of two theoretical explanations," Music Perception: An Interdisciplinary Journal, vol. 33, pp. 244–254, 12 2015.

[9] T. L. Bolton, "Rhythm." The American Journal of Psychology, vol. 6, no. 2, pp. 145–238, 1894.

[10] P. Fraisse, "6 – rhythm and tempo," Psychology of Music, pp. 149–180, 1982.

[11] Z. Lin, K. Xiao, and M. Riedl, "Generationmania: Learning to semantically choreograph," Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, vol. 15, no. 1, pp. 52–58, Oct. 2019.

[12] S. S. Stevens, J. Volkmann, and E. B. Newman, "A scale for the measurement

of the psychological magnitude pitch," The Journal of the Acoustical Society of America, vol. 8, no. 3, pp. 185–190, 1937.

[13] fraxtil. fraxtil's simfiles. Accessed: 25-Jan-2022.