MARKOVSYMPHONY - AN AUTOMATED MUSICAL PATTERN GENERATOR USING MARKOV MODELS

Francesco Piferi

Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB), Politecnico di Milano
Piazza Leonardo Da Vinci 32, 20122 Milano, Italy
francesco.piferi@mail.polimi.it

ABSTRACT

Automated music pattern generation is a burgeoning area of research that combines artificial intelligence and computer music composition. This article presents a novel system for creating music patterns using Markov Models, allowing for real-time playback via a built-in Sine Oscillator and MIDI export for use with virtual instruments. The system captures sequential dependencies within a given musical dataset and generates coherent yet diverse music patterns. Implemented in the JUCE framework, the system demonstrates its effectiveness in producing musical sequences and discusses potential applications in music composition and creative exploration.

***Index Terms*—** Automated Pattern Generator, Markov Models, Note Sequences

1. INTRODUCTION

Automated music pattern generation has emerged as a captivating field of study, where the fusion of artificial intelligence and music composition has unlocked unprecedented creative possibilities. In this article, it is introduced *MarkovSymphony*, a plugin developed on the JUCE framework [1], designed to harness the power of Markov Models for crafting musical patterns of variable length from a given sequence of notes. *MarkovSymphony* holds the key to transforming a sequence dataset into a Markov Model, enabling the system to accurately predict and generate compelling musical notes.

The ability to generate music patterns automatically has become a subject of immense interest due to its potential applications in various domains of music production, composition, and interactive experiences. *MarkovSymphony* addresses this demand by leveraging Markov Models, a powerful mathematical tool that captures the probabilistic dependencies between sequential events. By extracting salient patterns from the input sequence dataset, *MarkovSymphony* forges a unique musical pathway, unveiling the harmonious interplay between musical elements.

The central objective of *MarkovSymphony* is to provide a user-friendly and interactive platform for musicians, composers, and enthusiasts to unleash their creativity effortlessly. By accepting a sequence of notes as input, *MarkovSymphony* immerses itself in the realm of probabilities, learning the underlying musical relationships, and subsequently generating captivating patterns that resonate with the original sequence yet carry an air of novelty.

This article elucidates the inner workings of *MarkovSymphony*, delving into its architecture and the intricacies of the Markov Model implementation. The system's capacity for generating patterns of variable length grants artists the freedom to explore diverse musical expressions, whether in short, poignant sequences or in extensive, majestic symphonies.

It is also explore the real-time playback functionality of *MarkovSymphony*, employing a built-in Sine Oscillator to audibly experience the evolving music patterns. The instantaneous auditory feedback offered by this feature enables users to fine-tune the model's behaviour and align it with their artistic vision, fostering an immersive and iterative composition process.

Furthermore, *MarkovSymphony* extends its musical prowess by empowering users to export the generated sequences in the widely embraced MIDI format. This allows seamless integration with virtual instruments and digital audio workstations, where the full potential of the generated patterns can be unleashed, embellished, and expanded upon.

Throughout this article, it is demonstrated the effectiveness and versatility of *MarkovSymphony* through extensive experimentation and evaluation, by examining various users with age and musical backgrounds very different.

2. RELATED WORK

In this section it will be explore two different methods for automatic generation of musical patterns, *The Mozart Dice* [2] and *Impro-Visor* [3].

2.1. The Mozart Dice

The Mozart Dice [2] is a historically significant method of music composition attributed to Wolfgang Amadeus Mozart (27 January 1756 – 5 December 1791), one of the most renowned classical composers. This innovative approach, dating back to the 18th century, involves using a set of dice to generate musical phrases and sequences. Each face of the dice corresponds to a pre-composed musical fragment or motif, such as a melody, harmony, or rhythm. By rolling the dice, the composer could create novel combinations of musical elements, yielding unique and unexpected compositions.

Although not directly associated with Markov Models, *The Mozart dice* represents a pioneering effort in the field of automatic music composition using an iterative model. It stands as the initial attempt to explore stochastic and chance-based approaches for generating musical compositions, laying the foundation for future advancements in the field.

2.2. *Impro-Visor*: A Software for Jazz Improvisation based on Markov Models

In the domain of jazz improvisation, "*Impro-Visor*" [3] stands as a remarkable software tool that capitalises on the power of Markov Models. Developed as an intelligent music generation system, *Impro-Visor* caters to jazz musicians seeking spontaneous and expressive improvisational experiences.

The core concept of *Impro-Visor* revolves around capturing the essence of jazz idioms and stylistic patterns from a vast dataset of jazz performances. By analyzing these performances, *Impro-Visor* constructs a high-order Markov Model that encapsulates the intricate melodic, harmonic, and rhythmic relationships characteristic of jazz improvisation.

Upon receiving a user's musical input, such as a starting melody or chord progression, *Impro-Visor* harnesses the learned Markov Model to generate musically coherent and stylistically appropriate follow-up phrases and motifs. The result is a seamless and organic stream of improvisation, akin to a seasoned jazz musician responding to a given musical context in real-time.

Impro-Visor's integration of Markov Models into jazz improvisation introduces a novel dimension of spontaneity and creativity to traditional jazz performance. This software offers musicians an invaluable tool to explore new melodic pathways, experiment with different harmonic progressions, and engage in collaborative improvisations with the virtual ensemble.

3. MARKOV MODELS FOR MUSIC GENERATION

Music pattern generation has been an active area of research, drawing on various computational and statistical methods to model and generate musical sequences. Among these techniques, Markov Chains have proven to be a powerful and widely-used approach in music generation due to their ability to capture sequential dependencies within music patterns [4] [5]. In this section, it is provided an overview of relevant studies and approaches related to Music Pattern Generation using Markov Models.

3.1. Definition of Markov Model

Before diving into the discussion about the most relevant work on Markov Models let's see what actually it is.

Markov Models were initially theorised by the Russian mathematician Andrey Markov (14 June 1856 – 20 July 1922). They are stochastic processes that undergo transitions from one state to another. Over the years, they have found countless applications, especially for modelling processes and informing decision making, in the fields of physics, queuing theory, statistics and music.

It is also important to define the order of a Markov Model. It is the maximum number of samples that the chain can utilise in order to predict the following sample. For example, a first-order Markov Model will use only the last sample to predict the next one. A second-order Markov Model, will use the previous two samples and so on. Both first-order and second-order are depicted in Fig.1 and Fig.2 respectively, taking into account only 3 notes `(A, B, C)` and the following sequence to generate the Model: `ACBCABACBACA`.

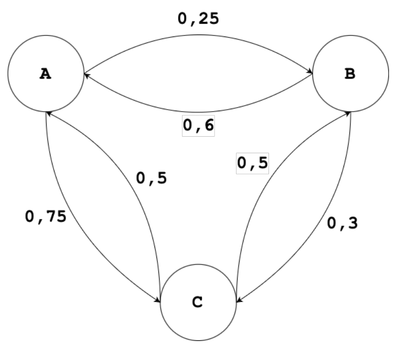

Figure 1: First Order Markov Model

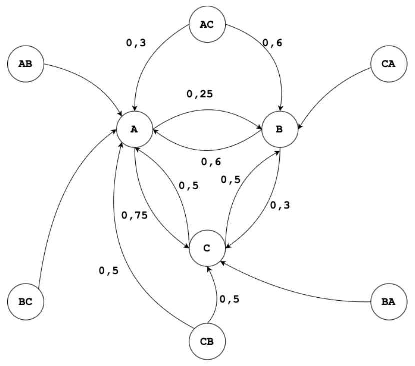

Figure 2: Second Order Markov Model

A pure Markov Model, at the beginning would choose the initial state randomly. In this project the first state is fixed by the user by pressing a note in his MIDI keyboard. The following states are automatically generated following the transitions of the Markov Model.

In this plugin's generation model, only the notes without the octave are considered. The decision to exclude the octave or MIDI numbers from the dataset was made to keep it manageable and user-friendly. Including the octave or MIDI numbers would have resulted in an excessively large dataset, making it challenging for users to provide input effectively.

By focusing solely on the notes, the plugin condenses the sequence within one octave and allows for the "propagation" of this information across all octaves. This approach simplifies the process of creating a new model for the user. Additionally, it introduces an element of randomness to the generated patterns, while still preserving the musicality of the sequences. This deliberate design choice strikes a balance between complexity and usability, enhancing the overall user experience.

3.2. Music Generation

As mentioned previously, the user begins the process by playing a note on his MIDI keyboard, which serves as the starting point for the Markov Model to predict a sequence of subsequent notes based on the plugin's knowledge. This knowledge can either be provided by the user or drawn from a preinstalled dataset, as explained in a dedicated section of the paper 3.3.

The user can choose the Markov Order, which determines how many previous samples the plugin considers for its predictions. When making predictions, the plugin seeks to use the maximum number of available samples, as long as they are equal to or less than the Markov Order selected by the user. If there are not enough samples to match the chosen Markov Order, the plugin adapts by decreasing the number of previous samples until it can make a valid prediction. However, the first prediction always employs a first-order Markov Model, and as more samples become available, the plugin gradually increases the order up to the user's maximum choice.

Moreover, the plugin's output is further shaped by another user-configurable parameter: the length of the generated sequence. Once all settings are configured, the plugin proceeds to play the generated sequence using an internal sine oscillator, allowing the user to gain an initial understanding of the melody's structure.

If the user find the newly generated pattern appealing, they can save it in MIDI format and seamlessly integrate it into a Digital Audio Workstation (DAW). This way, they can explore and experiment further, leveraging virtual instruments to create an enriched musical experience.

3.3. Dataset

The preinstalled dataset consists of the song *The Music of the Night* from the renowned musical *The Phantom of the Opera* composed by Andrew Lloyd Webber.

> «Andrew Lloyd Webber has composed the scores of some of the world's most famous musicals – including The Phantom of the Opera, Cats, Joseph and the Amazing Technicolor Dreamcoat, Evita, School of Rock, Sunset Boulevard and more. Being one of a small group in the world to receive EGOT status[1], Andrew is an industry titan and musical theatre giant.» [6]

This song was selected because I had complete access to the full notes of the song. It is essential to emphasise that having this particular song in the dataset does not mean that the generated song will be a replica of it or that every song leads to that song. Instead, the model is derived from the dataset, capturing the essence of the song and the relationships between the notes.

The process involves extracting information from the dataset to create a sequence of notes based on the model. As a result, the generated song may exhibit similarities to the original song, especially when a high Markov Order is utilized, and certain MIDI virtual instruments are employed. However, it is important to remember that the generated song is not intended to be a reproduction but rather an exploration of the underlying musical patterns and structures present in the dataset.

3.4. Example

A simplified sequence, `ACBCAB`, will be used to illustrate the music generation process. This sequence serves as the dataset for training the Markov Model.

[1]EGOT: acronym for the Emmy, Grammy, Oscar, and Tony Awards. Is the designation given to people who have won all four of the major American art awards.

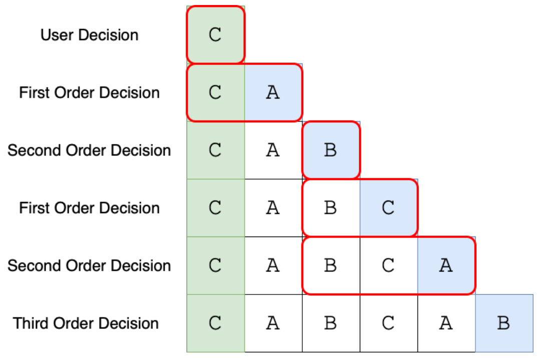

Figure 3: Process generation of the sequence `CABCAB`

3.4.1. *Generated Markov Model*

From the sequence it is possible to extract the Markov Chain. The following list is the output of the method `.getModelAsString()` of a Markov Chain Object. From it, it is possible deduce the transitions that will be evaluated in order to make a prediction. The structure of each line is the following:

```
Markov Order, Current State,:  Numbers of
possible next value, list of next values
```

```
    1,A,:2,C,B,
    1,B,:1,C,
    1,C,:2,B,A,
    2,A,C,:1,B,
    2,B,C,:1,A,
    2,C,A,:1,B,
    2,C,B,:1,C,
    3,A,C,B,:1,C,
    3,B,C,A,:1,B,
    3,C,B,C,:1,A,
...
```

As it is mentioned in 3.2, the algorithm is designed to maximise the value of the Markov Order. In Fig.3 it is represented the whole process to extract a new sequence. The boxes in red are the one taken into account to make the prediction.

4. GUI

The chosen framework for this project is JUCE, which is widely recognised as a leading platform for audio application and plug-in development.

> «JUCE is an open-source C++ codebase that offers compatibility with various operating systems, including Windows, macOS, Linux, iOS, and Android. It also supports the creation of VST, VST3, AU, AUv3, AAX, and LV2 plug-ins, ensuring broad accessibility and versatility for our music generation plugin.»[1]

To ensure a user-friendly experience, several `juce::Components` were utilized in building the plugin's graphical user interface. The interface consists of 3 `juce::Slider` elements, 4 `juce::TextButton` elements, and 1 `juce::ComboBox` element, thoughtfully designed to

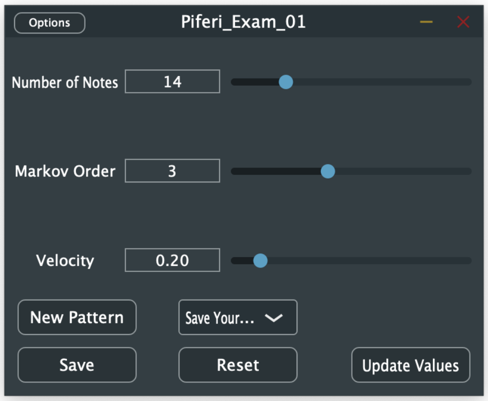

Figure 4: Graphical User Interface of *MarkovSymphony*

provide users with intuitive controls for seamless interaction and customization of their musical output. In the following list will be described one by one each component.

- `numberOfNotes`: this slider controls the length of the pattern that will be predicted by the Markov Model.
- `markovOrder`: this slider controls the depth of the Markov Model. The higher the order, the more precise will be the prediction.
- `duration`: this slider controls the duration of each note.
- `createButton`: this button empowers the user to initiate the creation of a fresh sequence, which will be utilised as the foundation for building a new Markov Model.
- `saveButton`: this button saves the new sequence and creates the new Markov Model.
- `reset`: this button resets all the component to their initial state. Also the Markov Model is reset to the preinstalled one.
- `saveButton` updates the Model with the values obtained from the sliders.
- `saveSequences`: this combo box saves the selected sequence in a MIDI file.

5. MIDI EXPORTATION

A pivotal aspect of the plugin is its MIDI exportation capability, allowing users to save the generated music sequence from the Markov Model as a MIDI file. This feature holds immense value as it enables users to import the generated melody directly into their preferred Digital Audio Workstation (DAW). By doing so, users can seamlessly integrate the generated sequence as a dedicated track within their music production projects, enhancing their creative capabilities and streamlining the composition process.

The MIDI file can be saved simply clicking on the combo box. Once the combo is open it will show a list of 10 items as depicted in Fig.5. Those items are the last 10 melodies generated by the plugin. Clicking on one of them it will automatically save the `.mid` in an external `output` file.

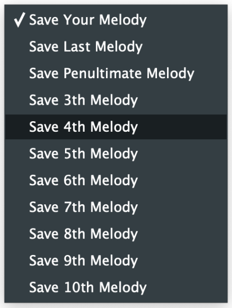

Figure 5: Combo Box items

6. USER EVALUATION

The study involved a total of 12 diverse users who participated in a User Evaluation of the plugin. Following a brief explanation of the plugin's features, the users were given the opportunity to interact with it. The participant group was intentionally diverse, comprising both musicians and non-musicians, with varying levels of musical expertise. Specifically, 2 participants were conservatory students, 6 were individuals who either currently play or have past experience with at least one instrument, and the remaining 4 had no prior experience playing any musical instrument. To distinguish among the participants easily, they were denoted as User1, User2, User3 and so forth, in the order listed below:

- User1, User2: Conservatory students.
- User3, ..., User8: Play or used to play an instrument.
- User9, ..., User12: Never played an instrument.

Each participant was invited to provide feedback by answering three resource questions. Alongside each question, they were asked to complete a survey where they rated the *musicality* of the generated output. The survey required them to assess the auditory experience using categories such as *very bad*, *bad*, *good*, *very good*, or *awesome*. The graphical representations presented below illustrate the results of these surveys, illustrating the participants' evaluations.

The resource questions were as follows:

- **RQ1**: After using the plugin with its default settings, what are your thoughts on the *musicality* of the played notes?
- **RQ2**: How does adjusting the value of the Markov Order impact the *musicality* of the generated sequences?
- **RQ3**: Experiment with creating your own sequence by clicking on *New Sequence* and then *Save*. What do you think about the new sequences you generate? Are they similar to the one you provided?
- **RQ3b**: How does changing the value of the Markov Order affect the *musicality* of your generated Markov Model sequences?

Starting from the first Resource Question I will briefly expose the result of the answer and give an interpretation of the survey results. For all the evaluations it was used GarageBand as external DAW. Each test was run on the computer keyboard because I was unable to conduct the experiment on an external MIDI keyboard.

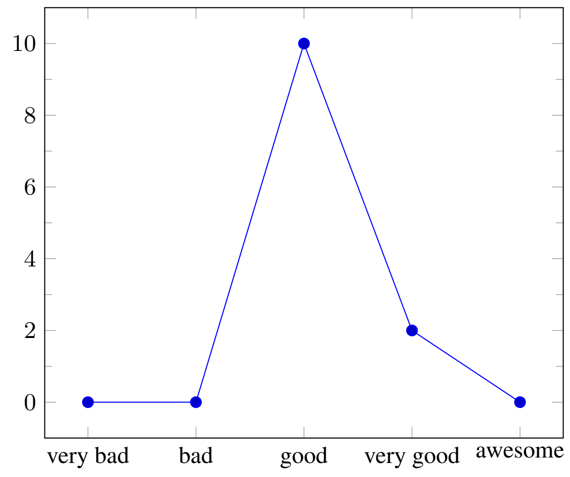

Figure 6: Resource Question 1

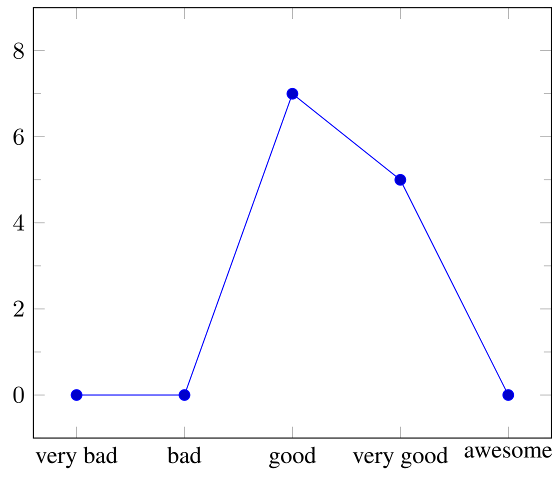

Figure 7: Resource Question 2

This could have affected the test, particularly the final two questions when playing a series of notes was required.

6.1. Resource Question 1

During the initial evaluation phase, participants were introduced to the active plugin within an external DAW. After becoming acquainted with the plugin's functionalities through a brief familiarisation period, they were presented with a specific question.

The responses received demonstrated a remarkable consensus, with participants consistently expressing their satisfaction with the pleasing auditory experience offered by the notes generated through the plugin. This sentiment held true not only during direct playback from the plugin itself but also when utilising a virtual instrument after exporting the generated MIDI file. Notably, none of the participants regarded the generated sequence as "terrible," nor did anyone classify it as "awesome." Among the participants, 10 rated the experience 3 out of 5, while 2 participants awarded it a 4 out of 5 rating, as it is possible to see in Fig. 6.

6.2. Resource Question 2

For the second question, participants were tasked with adjusting the Markov Order using the second slider. This step prompted an intriguing examination of nuanced distinctions within the responses.

Interestingly, Users 5, 10, and 12 noted no perceptible difference compared to the prior configuration. This observation was substantiated by their consistent rating, which remained unchanged at 3 out of 5.

In contrast, Users 2, 4, 6, and 7 highlighted the impact of higher Markov Orders, such as 5 or 6, on the sequence's repetition. These participants observed that with lower Markov Order values, the generated melody exhibited greater creativity. This phenomenon aligns with theoretical expectations: a Markov Order of 1 implies that the succeeding note depends solely on the current note, whereas a higher Markov Order like 5 or 6 introduces multiple preceding notes, leading to statistically similar subsequent states.

Collectively, the feedback received reflects a noticeable level of satisfaction among participants, manifesting through modest yet consistent enhancements in their evaluations.

6.3. Resource Question 3

As one might anticipate, this question emerged as the most divisive among participant responses. Following an initial attempt, a majority of participants produced random note sequences, resulting in less favorable generation outcomes. Consequently, two participants, specifically User 6 and User 8, provided negative ratings (2 out of 5).

On the other hand, other participants engaged in diverse experimentation before evaluating the plugin. A notable majority, in particular Users 2, 3, 4, 5, 7, 10, 11, and 12, opted for a C Ionian Scale (`C D E F G A B C`), varying between `C3` and `C4`, reversing directions, or performing bidirectional sequences. These explorations yielded significantly improved plugin outputs, as evident in Fig. 8.

Another avenue for personalizing the Markov Model involved playing simple melodies. User 1, 2, 5, and 9 adopted this approach, each selecting a distinct piece:

- User 1: *Für Elise* by Ludwig van Beethoven
- User 2: *Symphony No. 9* by Ludwig van Beethoven
- User 5 and User 9: *Happy Birthday To You*

Remarkably, in each of these instances, the generated outputs were well-received, prompting a positive 5 out of 5 evaluation for User 1's case.

6.4. Resource Question 3b

The responses to this Resource Question are similar to those discussed in Section 6.2. A discernible trend emerged where simpler model configurations facilitated the generation of analogous melodies. For instance, if a C Ionian Scale is chosen, spanning from `C3` to `C4`, the resultant model would inherently replicate the selected scale, commencing from the triggered note. As the model complexity heightened, the generated sequences embraced greater

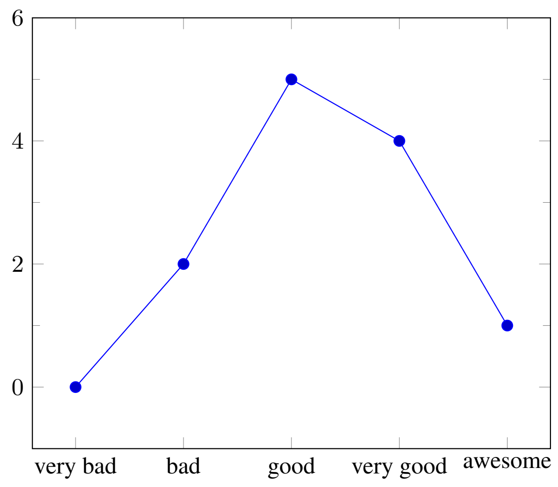

Figure 8: Resource Question 3

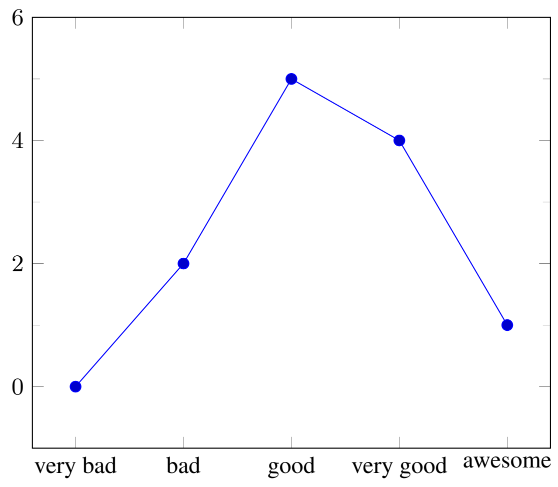

Figure 9: Resource Question 3b

originality. The evaluations exhibited stability even subsequent to modifications in the Markov Order as shown in Fig. 9.

7. FUTURE IMPROVEMENTS

The field of music sequence generators has been evolving with a constant drive for improvement. Future directions could include investigating Artificial Intelligence (AI) systems as an alternative to conventional Markov Models. AI presents the fascinating possibility of uncovering complex relationships within constrained musical sequences, expanding the breadth and depth of generated compositions.

Additionally, a promising direction is the incorporation of external models into the existing framework. Users might be able to import various models through this integration, bringing a variety of musical perspectives into their creative process.

A noteworthy idea is also to take into account integrating an external MIDI keyboard for testing. By avoiding the constraints imposed by input from a computer keyboard, this inclusion might offer a more genuine evaluation process. An external keyboard could be added to the testing setup to allow for a more thorough evaluation of the system's responsiveness and adaptability to live musical input.

8. CONCLUSION

this study presented *MarkovSymphony*, an innovative plugin developed within the JUCE framework, showcasing its proficiency in automated music pattern generation through Markov Models. The plugin's real-time sequence generation capabilities, complemented by MIDI export functionality, underscore its potential as a versatile tool for musicians and composers.

The empirical evaluation shed light on intriguing insights, revealing a dynamic interplay between Markov Order adjustments and the generated musical sequences. Participants' feedback elucidated a clear preference for balanced Markov Orders, where creative melodies coexist harmoniously with cohesive musical structures.

However, the study acknowledges limitations, such as the use of a computer keyboard for testing, which might have influenced the evaluation outcomes. Future enhancements could involve incorporating AI approaches for more intricate sequence modelling and enabling external model integration. Integrating external MIDI keyboards for testing could yield more authentic results, enhancing the plugin's real-world usability.

9. ACKNOWLEDGEMENT

This project marks the conclusion of my exams at Politecnico di Milano. For this reason I would like to thank each friend, student, professor and family member who has supported me over the past five years..

A special thank to each participant of the evaluation for its invaluable contributions and valuable insights that have significantly enriched the project.

10. REFERENCES

[1] "JUCE Official Site," https://juce.com.

[2] Z. Ruttkay, "Composing mozart variations with dice," *Teaching Statistics*, vol. 19, no. 1, pp. 18–19, 1997.

[3] M. Zanoni, "Grammar- and Automata-based agents, slides 29-36," Politecnico di Milano, 2022.

[4] A. Yanchenko, "Classical Music Composition Using Hidden Markov Models," https://dukespace.lib.duke.edu/dspace/handle/10161/15245, 2017.

[5] J. Cruz, "Deep learning vs markov model in music generation," Honors College, 2019.

[6] "Andrew Lloyd Webber's Official Website," https://www.andrewlloydwebber.com.