MELODY GENERATION WITH MARKOV MODELS, A RULE BASED APPROACH

Michael Finley
University of Rochester
mfinley4@ur.rochester.edu

Gavin Baker
University of Rochester
gbaker8@ur.rochester.edu

ABSTRACT

Algorithmic composition is an important area of musical research in machine learning and artificial intelligence. We propose a low-complexity solution to melody generation that develops melodies based on expert rule-based models and learned melody generation from Markov Models.

Recent advances in machine learning based techniques for algorithmic composition have created an excellent set of methods for generating artistic musical works. These methods, while intrinsically very different from rule based models for the same purpose, can produce similar outputs if both are executed well. This project compares and contrasts a rule-based model and a markov model for melody generation. Both models strive to compare to a dataset of folk songs parsed into MIDI files, outputting MIDI data for playback and comparison. We compare the model outputs both quantitatively and qualitatively, both in terms of loyalty to source material, and in terms of adherence to music theory rules and practices.

Our comparative results confirm the comparability of our melody generation with two additional advantages being flexibility of melody generation based on user preference and user dataset selection to program in inherent biases.

1. INTRODUCTION

Automated music generation, or Computer Music, has been present in computing research since computers were able to communicate with musical instruments- as early as the 1950s researchers were developing algorithmic composition methods to varying degrees of success. Only recently, however, due to the boom in machine learning technology and methods, has the field of computer music and algorithmic composition been able to rapidly generate large sets of music that could have plausibly been produced by a human composer.

Combining melody generation with real-time musical key estimation [4] and chord identification can lead to real-time accompaniment and improvisation. Real-time accompaniment and musical improvisation is one of the many abstract goals of machine learning in the context of music. A sophisticated and simple melody generation system can aid in this pursuit, similar to the one we have developed in this project.

Rule based models of algorithmic composition keep true to the title of the field- they use explicitly defined algorithms to generate melodies and pieces of music according to rules. These rules can range anywhere from the CHORAL system written by K. Ebcioglu for deterministically harmonizing Bach chorales [2], to K. Burns, who used stochastic processes in order to facilitate melody generation coupled with a set of rules [1]. These methods were the prevalent methods of algorithmic composition until machine learning methods took over a large part of the computer music field in the 2000s and 2010s.

Machine learning based methods have revolutionized the algorithmic composition field. Dong et al. used Generative Adversarial Networks (GANs) to enforce conformity to a dataset by a neural network both monophonically and polyphonically [3]. Kumar et al. used LSTM architecture-based recurrent neural networks (RNNs) for the same purpose, this time taking broader melodic structure over time into account [7]. These methods are not only significantly different from rule-based computer models of music, they also are much more easily generalizable to different musical genres, moods, and many other differentiating features that previously would have required entirely new algorithms for rule-based music generation.

In addition to the various neural network based machine learning algorithmic composition methods, there are many other methods that also produce high quality results. W. Schulze uses a hidden markov model to keep track of both note output and long term melodic structure in [8], which results in melodies that not only keep track of melodic structure over time, but also allows for short-term expressions of smaller melodic fragments, while maintaining a smaller model footprint than would be created with a neural network based model.

This project aims to compare, both qualitatively and quantitatively, the musical melodies generated by both rule-based models and machine learning based models for algorithmic composition. To this end, we have developed both a rule-based model for melody generation, as well as a markov model for the same purpose. We compare and contrast the two models in terms of complexity, as well as the musical quality of the outputs of the models.

2. DATASET

The dataset we use in our system development is a subset of The Meertens Tune Collections [6] which is comprised of a diverse set of musical data. The datasets utilized for training were MTC-LC-1.0, MTC-ANN-1.0, MTC-ANN-

2.0.1, MTC-FS-INST-2.0 and MTC-INST-1.0 comprising a total of 23,190 melodies used in various publications. The primary dataset, MTC-FS-INST-2.0, were comprised of melodies collected from Dutch sources from five centuries, including prints, manuscripts, and field recordings. The metadata includes: source, tune familiy membership, composer, segmentation at phrase level, key, meter, dating, and geotags. The metadata of interest is only the key and meter so as to transpose the melodies to any meter or key when being used for training. The system we have developed allows the user to train agnostic to key or meter or not to allow for more specific melody generation depending on user interest.

This dataset is one of many publicly available, annotated, midi melody datasets that suits the purpose of being a proof-of-concept dataset. The inherent bias of the dataset is understood to affect the resulting melody generation of our system. For a large-scale industry implementation one would want to collect a more robust, diverse, dataset for training. Some data preparation infrastructure is required to interface with the dataset for training purposes depending on what platform is used to implement the system.

3. METHOD

Melody generation can be done in both the rule-based approach and the learned approach. The rule-based approach needs to dataset for training and requires only a set of rules to determine the transition probabilities of melody generation transition matrix.

The learned approach requires a dataset to develop the transition probabilities for the Markov process. The dataset the user utilizes for transition probability training will determine what biases the trained model develops. As such, great care should be taken upon the user as to which datasets are used depending on their goals of melody generation when using this system. The dataset that is used for training has the largest effect on results. The following subsections will detail the mathematical basis for both the rule-based process and the Markov Model approach.

3.1 Rule-based Approach

The rule-based approach employs expert music theorist knowledge to develop a set of rules by which a melody can be algorithmically composed. The rule-based model employs a progressive embellishment approach to constructing a melody, iteratively refining a melody according to user-specified parameters. No data-based learning occurs in this version of the melody-generating model.

The system outputs a melody by a multi-step process. This process begins by setting up a basic 'chord progression' determined one of two processes, depending on the desired output. The first process takes an input melody and 'flattens' it into a broad contour based on chords and most frequently played note. This method allows the model to 'improvise' over a given base melody. Alternatively, the system can simply produce a chord progression for itself by invoking a set of rules in a randomly determined order. These rules include stipulations that the chord progression end on the IV or V chord, that it start on the I chord, and that each movement downward should cause the progression to trend back up, and vice versa. After this step is complete, the model has created an initial melody that can now be refined by the progressive embellishment approach.

The progressive embellishment technique takes in a sequence of notes, and with probability distribution described in the next paragraph per note, splits that note into two equally valued notes (e.g. one whole note splits into two half notes), each with different but related pitches. The model has six choices for which set of notes to split a note into. These sets both raise and lower the embellished notes, and allow for greater range of both pitch range and also note value range.

As was previously noted, the probability of a note being embellished changes as the note changes in value. A longer note is more likely to be split when it is longer, and less so when it is shorter. As a function of value (in fractions of a measure, i.e. a whole note has value 1), the probability of being embellished is:

$$P = .8\sqrt{value} \tag{1}$$

The rule-based approach allows for a user to determine to what extent certain qualities of a melody should be adhered to. The changeable qualities we explore in our system are intricacy, complexity, and tonal center/key. Intricacy and complexity are represented on a scale from 1-10, while tonal center and key have the domain of all possible MIDI notes. The intricacy parameter controls how many rounds of progressive embellishment are completed, and thus how 'intricate' and detailed the melody is. The complexity parameter determines the length and complexity of the initial chord progression created in the first step of the process. Tonal center and key parameters allow for the shifting of the starting pitch and mode of the melody.

Once the user has selected the qualities of the melody they desire, the model computes random seeds for all probability distributions, and then runs the specified number of iterations of each portion of the model, computing an output melody with the desired parameters.

3.2 Markov Model Approach

The learned approach employs a learning approach to develop transition probabilities for melody generation. A Markov process is defined as a process that satisfies the Markov property, where a system can only make decisions about its future states based solely on its current state. This property is often referred to as memoryless-ness. The Markov Model process can be understood from the diagram in Fig. 2.

The Markov stochastic process is desirable in melody generation for many reasons. One reason being that once these Markov processes are generalized to be n-gram Markov Models, the system learns structures from the melodies that it learns from. These structures end up sounding like building blocks for melodies where a certain

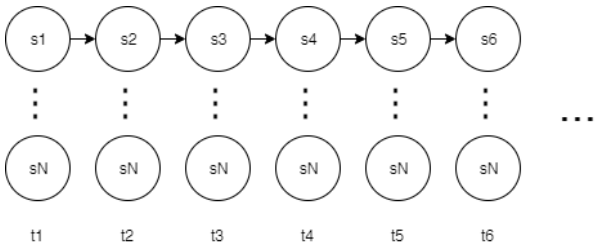

Figure 1. State diagram snapped to a time grid determined by tempo and duration of desired melody.

melodic phrase has a higher probability of appearing than other musical notes. We define our Markov Model to satisfy the following property for a sequence X_n, defined in Eqn (2).

$$Pr(X_{n+1} = x|X_1 = x_1, ..., X_n = x_n) = Pr(X_{n+1} = x|X_{n-3} = x_{n-3}, ..., X_{n-1} = x_{n-1}) \quad (2)$$

The system can be trained on whatever dataset the user desires. For a proof of concept, the MTC dataset was used. Upon complete training of the transition probabilities, melodies can be generated with some level of control over meter and duration in a similar fashion to the rule-based model. A convenient diagram showcasing the structure of a transition matrix can be seen in Fig. 1.

1st-order matrix

Note	A	C♯	E♭
A	0.1	0.6	0.3
C♯	0.25	0.05	0.7
E♭	0.7	0.3	0

2nd-order matrix

Notes	A	D	G
AA	0.18	0.6	0.22
AD	0.5	0.5	0
AG	0.15	0.75	0.1
DD	0	0	1
DA	0.25	0	0.75
DG	0.9	0.1	0
GG	0.4	0.4	0.2
GA	0.5	0.25	0.25
GD	1	0	0

Figure 2. Rank one Markov Model, and a rank two Markov Model transition matrix.

Fig. 1 demonstrates the increase in transition probabilities as the rank of the Markov Model increases. It was found that the increase in melodic phrase capturing begins to decrease significantly above third rank Markov Models. As such, we developed our system to be a tri-gram Markov Model for generating melodies based on learned data. This finding is consistent with findings from studies in speech recognition, which find that tri-gram Markov models are similarly best suited to understanding speech patterns [5].

4. RESULTS

The results of both models were promising. Collection of a subjective metric for melodic pleasure and musicality is needed for further results and development of the system. Our markov model was trained on Dutch folk melodies which come from a distinctly different musical lineage from what most people in our scientific and musical community are exposed to. As a result, we were unable to collect subjective impressions and results of the melodies our system was able to generate. Additionally, the markov model generated melodies that, while preserving musical phrasing of the original song, had very little regard for tonal center or ability to resolve cleanly to important notes in the key. The melodies generated, however, did accurately carry over traits and phrasings from the training data, however.

In contrast, the rule-based approach retained none of the phrasing details that the markov model was able to learn. However, it was able to preserve the overall contour of the melody very well, and also was more able to adhere to best music theory practices.

We feel a combination of the rule-based approach and the learned model would result in the most robust system from melody generation. This hybrid system would allow the user to fine-tune their desired melody generation while still taking into account melodic phrases which can be learned from a variety of datasets assuming the proper data preparation is done.

5. CONCLUSION

The rule-based and markov models each succeed at very different parts of melody generation. It is not possible to generalize one model as superior to the other, as they each have both strengths and shortcomings.

Due to the limitations of the rule based model in terms of ability to adhere to its source material and the implementation of the smaller scale melodic phrases, the rule based model produces melodies that conform to its rules, but do not allow the generated melody to be more than loosely based off of the source melodies. The hidden markov model, on the other hand, allows for much better performance in terms of artistic imitation between training data and the model output.

We find that the optimal model for generating melodies based on source data would be a marriage of these two models, combining the music theory adherence of the rule-based model with the motif-level imitation of training data of the markov model.

Since these methods produce output that must be evaluated by a human, it is difficult to quantify the effectiveness of the methods we employed in this project to the methods developed by other projects working on the same project. Given that the standard is similar in that humans either judge or create the methods of judgement for melodies based on artistic preference, however, we can assert that our methods produce results analogous to many other machine learning based methods.

5.1 Future Work

The clear next step in this work is the building of a hybrid model- evaluating which portions of each of the models shown in this work contribute to its own strengths and drawbacks, and fusing the most advantageous parts together to produce a melody generator with all of the successes of each model- the theory of the rule-based model and the expressiveness of the markov model.

Additionally, while independent melody generation is a pursuit with standalone merits, a logical extension of this work is the inclusion of real time user-defined musical input, to which the model would play alongside. While this would force the inclusion of many more parameters and degrees of freedom into the computations in the model for upcoming outputted notes, modern computing hardware and software optimizations makes it more than feasible in a real time scenario. Similarly, one could train the markov model on chords instead of notes, and then play along in a similar fashion with the same amount of input or even possibly fewer degrees of freedom.

6. REFERENCES

[1] K. Burns. Algorithmic composition, a definition, 1997.

[2] Kemal Ebcioğlu. An expert system for harmonizing chorales in the style of j.s. bach. *The Journal of Logic Programming*, 8(1):145 – 185, 1990. Special Issue: Logic Programming Applications.

[3] H.W. Dong et al. Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment, 2017.

[4] M. Finley and A. Razi. Musical key estimation with unsupervised pattern recognition. In *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0401–0408, Jan 2019.

[5] Mark Gales and Steve Young. The application of hidden markov models in speech recognition, January 2007.

[6] Meertens Institute. The meertens tune collections. In *Proceedings of the International Symposium on Music Information Retrieval*, 2019.

[7] Harish Kumar and Balaraman Ravindran. Polyphonic music composition with LSTM neural networks and reinforcement learning. *CoRR*, abs/1902.01973, 2019.

[8] W. Schulze. Music generation with markov models. *IEEE Multimedia*, 1(3):1–6, 2007.