Music Generation with HMMs and RNNs

Chuan Chen
Center for Data Science
New York University
New York, NY 10003
cc6580@nyu.edu

Yanqi Xu
Center for Data Science
New York University
New York, NY 10003
yx2105@nyu.edu

1 Introduction

Computational music generation has long been the problem of interest in many music theory and computer science researches. This is mainly because the intrinsic mathematical and statistical properties of music as time series sequences provide intuitions for music modeling. Before the recent development in deep learning and neural networks, many of the music generation algorithms have been based on hidden Markov models. The basic structure of this class of probabilistic models resonates with the music composition process by humans: the hidden states of HMMs represent a certain underlying abstract state of mind of the composers and as the hidden states progress in certain patterns, they give rise to the music notes. Moreover, a deep markov approach has been recently proposed for music generation for its ability to model both nonlinear and linear relationship between hidden states. [KSS16]. Besides HMM and deep markov, Recurrent Neural Networks (RNN) also possess great modeling power over sequential time series data such as music. Specifically, a robust version called the Long Short-Term Memory networks have proven to yield great results in interpreting real world data [KY18].

In this paper, we will compare the HMMs probabilistic approaches with a deep learning approach towards music composition. We want to know, with the least amount of music theory knowledge, which approach generates better music in terms of their long-term melodic and short-term harmonic structures. We will be focusing on hidden Markov models and its variants as well as long short term memory networks (LSTM) and deep markov models for generating music from 384 Bach's Chorales. We compared and evaluated the music generated by different models based on their originality, musicality, and temporal structure. Our results show that the two deep learning models generally out performs the HMM models, with the LSTM model far exceeding the DMM in learning the distribution of harmony/melody of the original music and capturing long-term temporal structure.

2 Related Work

Back in late 90s, Marom used a hidden Markov model for jazz improvisation where he treated the drum track from the jazz ensemble as the hidden states, which controls the interactions among multiple components in a piece of music[Mar97]. More recently, Wysocki and Graci created simple melodic songs with HMMs[WG15]. In 2017, Yanchenko trained HMM variants with piano pieces from the Romantic era to generate new music in a similar style[YM17]. LSTMs were first introduced in 1997, and about 10 years later, LSTMs started to revolutionize speech recognition. It's application is significantly broadened in recent years especially in the field of music interpretation/generation. It became a building stone for one of the most famous music and art machine learning research projects, Magenta by Google AI [Cai18]. Recently, many individual projects have also explored in using LSTM for music generation, such as Paul and Nikhil's paper published in 2018.

This year, Cruz in his honor thesis used both HMMs and Biaxial RNN in producing music that resembles the original composers[Cru19]. Then a classifier were applied to the music to determine which technique generates music that is correctly classified as being from the composer they were

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

trained on. The Cruz paper concluded that deep learning approach creates music with higher accuracy in the classifier. Our paper also performs a HMM and RNN comparison, but our approach slightly differs from Cruz's in two aspects: 1) We created models on both simple HMMs and some HMM variants with hierarchical structures to deeper explore the potential of HMM structures; 2) Instead of using a classifier, we evaluated the generated music analytically based on three different metrics.

3 Problem Definition and Algorithm

3.1 Task

The task of our research is to compare the performance of various HMMs, LSTM networks, and DMM in learning from 382 fourpart harmonized chorales by J. S. Bach and generating new music form them. The input of all the models are the notes of the music pieces converted into either a 1-D sequence or piano-roll vector sequence to better feed into HMMs or LSTMs. The output are the generated music in midi format, which can be converted to audio for listening.

3.2 Algorithms

In our research, we altogether explored six different models for music generation. A simple HMM was first fitted as our baseline and three HMM variants were examined including two-layered HMM, two-layered autoregressive HMM [YM17] and hierarchical HMM . (Hierarchical HMMs are excluded from the paper since we failed to implement it from scratch and thus no results were generated by this type of models.) We chose long short-term memory networks as our deep learning approach for its ability to capture global structure of sequential data. Finally, deep Markov model, a model combining the structure of both HMM and neural Network, was examined for its ability to model nonlinear relations between different states[KSS16].

3.2.1 Simple HMM

Hidden Markov models contain two types of discrete variables. The hidden states are unobserved Markov process following Markov properties. Each hidden state emits observations with a multinomial distribution. The structure of a HMM is completely captured by its initial distribution π, transition matrix A and emission matrix C. The graphical structure of a hidden Markov model is displayed in Figure 1.

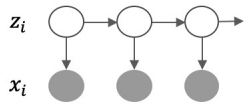

Figure 1: Graphical Model of HMM

For our case, the music notes are observations and the number of hidden states are fixed as a hyperparameter. In order to model piano notes as a linear sequence, we adopted the music annotation strategy proposed in the Yanchenko paper[YM17]. We directly modelled the sequential notes from midi files. Figure 2 illustrates how midi format represent music note as a linear sequence of data with an example of Greensleeves.

There are 88 different keys on a piano, so presumably there are 88 observation states at most. However, most music only uses part of the notes. Hence, the number of unique piano notes appears in a music is the number of observation states. In our example, the number of observation states is 53. We trained three models with 10, 20 and 30 hidden states respectively. We initialized initial probability, transition matrix and emission matrix with random numbers. A forward-backward algorithm are applied to find the posterior probabilities and we use EM algorithm to find the optimal parameters for the model. The new music is sampled from the model with the learned parameters.

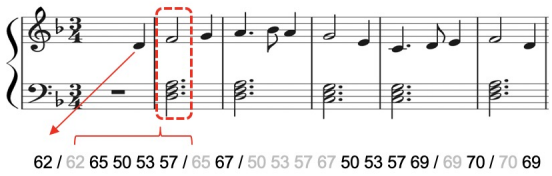

Figure 2: A illustration of MIDI Music Annotation The first three bars of the sheet music are represented by the sequence of numbers underneath. Each group separated by slashes represents each timestamp. It's known that music notes can be represented by numbers in Music Theory. The color grey indicates the this notes are turned off at this timestamp and Black indicates the notes are turned on. The second group 62 65 50 53 57 can be interpreted as "The note 62 is turned off and notes 65 50 53 57 are played at this timestamp."

3.2.2 Two-layered HMM

Two-layered HMM is a simple HMM with an additional layer of hidden states that govern the behaviors of the original hidden states. The graphic structure of a two-layered HMM is shown on the left side of Figure 3. The middle layer can be treated as both the observation states of the upper layer and the hidden states of the lower layer observation states. This type of HMM with certain hierarchical structures are used with the hope to better model the global structure of music pieces. The inference and learning algorithms for a two-layered HMM are relatively simple since we can model the interactions between each layers separately (Figure 3). The set of parameters of the model is $\Omega = \{\pi, A, C, \pi', A', C'\}$ where π', A' and C' denote the initial probabilities, transition matrix and emission matrix between y_i's and z_i's. The procedures for parameter learning are listed below:

1. The bottom two layers x_i and z_i can be seen as a simple Markov model and we can apply standard inference and learning algorithms to learn the parameters π, A and C.
2. Find the best possible sequence for the hidden states z_i with Viterbi algorithm.
3. Set the best possible sequence to be the z_i and treat z_i as the observation states and the upper layer y_i are the hidden states. Then apply standard inference and learning algorithms to learn the parameters π', A' and C'.

We trained the model with 10 and 20 hidden states for both parts, due to the fact that more hidden states might be more difficult to learn the parameters.

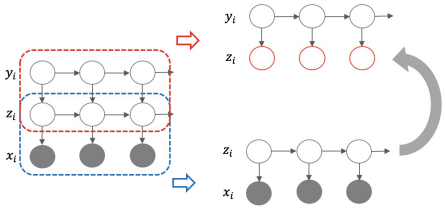

Figure 3: THe Graphical Model of LHMM

3.2.3 Two-layered Autoregressive HMM

A two-layered Autoregressive HMM is a layered HMM with a dependence directly added on the observations to simulate the dependence structures between each music notes. The parameter learning procedure is the same as LHMM's described above, except for that HMM of the first part is autoregressive. The joint distribution of a autoregressive HMM is

$$P(z_{1:t}, x_{1:t}) = P(z_1)P(x_1|z_1)\prod_{i=2}^{t} P(z_i|z_{i-1})P(x_i|x_{i-1}, z_i).$$

Hence, α and β can be written as

$$\alpha(z_i) = P(z_i, x_{1:i}) = P(x_i|z_i, x_{i-1}\sum_{z_{i-1}} \alpha(z_{i-1})P(z_i|z_{i-1}).$$

$$\beta(z_i) = P(x_{i+1:t}|z_i, x_i) = \sum_{z_{i+1}} \beta(z_{i+1}) P(x_{i+1}|z_{i+1}, x_i) P(z_{i+1}|z_i).$$

in an autogressive forward-backward inference algorithm.

We trained a first-order autoregressive two-layered hidden Markov model with 5 hidden states because it took too much time for inference due its complex strucutres.

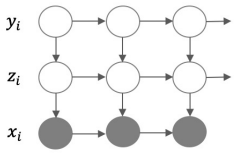

Figure 4: The Graphical Model of Two-layered Autoregressive HMM

3.2.4 LSTM

Recurrent Neural Networks are very popular deep learning models that are used to carry out many different deep learning tasks such as time series analysis and machine interpretation/generations of real world data. Since RNNs are recurrent in nature, they preserve internal memory. Figure 5 shows the graphical model of an unrolled RNN, and it is apparent that the output of each previous input is also carried down as the input for the next step

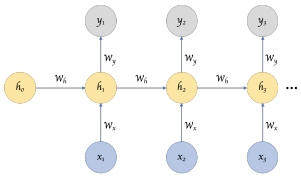

Figure 5: An unrolled recurrent neural network

Long Short-Term memory networks are a modification of RNN's that resolves the computational limit of vanishing gradients, which results in decreased learning, while also improving its ability to pass down memory. LSTMs control information flow by the inclusion of a forget gate that uses a sigmoid layer to discard unimportant information, an input gate that regulates which inputs to pass down the way, and an output gate that outputs the next hidden state by first running a sigmoid layer and then tanh.

Before implementing the LSTM model, music files are read using the Music21 Python toolkit into the following object types where each object contains information about the pitch, octave, and offset of notes in the music [Music21]. A sequence is formed using all the notes from the music dataset and

```
...
<music21.note.Note B> 72.0
<music21.chord.Chord E3 A3> 72.0
<music21.note.Note A> 72.5
```

Figure 6: Music21 annotation

encoded as numerical data and used to create inputs and outputs for the model. We then constructed a LSTM model with the layers shown in figure 7 with regularisation set as 0.1 to avoid over-fitting and the softmax activation function. The model was trained and the weights with the lowest categorical cross entropy loss from 77 epochs were used as weights for music generation.

Layer (type)	Output Shape	Param #
lstm_1 (LSTM)	(None, 100, 512)	1052672
dropout_1 (Dropout)	(None, 100, 512)	0
lstm_2 (LSTM)	(None, 100, 512)	2099200
dropout_2 (Dropout)	(None, 100, 512)	0
lstm_3 (LSTM)	(None, 512)	2099200
dense_1 (Dense)	(None, 256)	131328
dropout_3 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 456)	117192
activation_1 (Activation)	(None, 456)	0

Figure 7: Structure of LSTM

3.2.5 Deep Markov Model

At last, we discovered a special class of generative models called deep Markov models. The major difference between traditional hidden Markov models and deep Markov models is the linear transition and emission distributions are replaced by neural networks to model the nonlinear complex relations inside the matrices. The overall architecture of the deep markov model we used for music generation is shown in Figure 8. The observations of our model is a sequence of binary vectors with length 88 to indicate which piano key is played. The model uses a three layer neural network to parameterize the emission distributions. The first two layers are linear transformations that convert the dimension of a hidden state into a vector of length 88. Then a sigmoid layer is applied after that to transform each element of the vector into valid probabilities. In this way, the emission probabilities are parameterized into Bernoulli distributions. The hidden states are vectors of length 200 and are modeled as multivariate Gaussian distributions. Hence, three layer neural networks are applied to both the mean and covariance of the transition distributions. In this part, the model uses a gated unit to weight the linear and nonlinear parts of the transition function, which allows some dimensions to have a linear transitions while others have non-linear transitions.

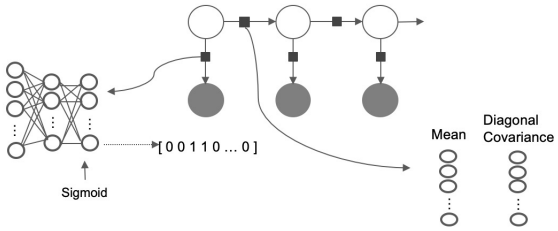

Figure 8: Deep Markov Model

This class of models usually uses variational inference for parameter learning. Variational inference is a type of inference algorithm approximates intractable probability densities with optimization. In our research, we treated this part as a black box due to time limit. We make the assumption that the parameters learned with our chosen algorithms are optimal and has nondeteministic affects to our final results.

4 Experimental Evaluation

4.1 Data

The dataset we used throughout our research is *JSB chorales* [BBV12], which contains the entire corpus of 382 fourpart harmonized chorales by J. S. Bach. The data are already formatted as pianoroll matrices to directly feed into LSTM and deep Markov models. When training HMMs, the pickle file was converted into 1-D linear sequential data according to the method described in above Figure 2.

4.2 Metrics

Although music is subjective, and an absolute evaluation of better or worse between different pieces can not be obtained, there are certain quantitative music characteristics that we can measure. Specifically, we used three different metrics to evaluate across the generated music from our models, originality, musicality, and temporal structure.

From his paper "Measuring Musical Originality Using Information Theory", Coffman proposed using the entropy of note sequences as a measure of originality [Cof92]. Entropy is a measure of information of a random variable based on its probability distribution, and it can be used to measure how predictable a musical piece is. Disregarding musical coherence, a high entropy can represent high originality in a piece of music. Thus, we will be calculating the entropy of music generated from each model and see which model produces more original music.

Music generated from a good model should also capture the musical style of a particular genre or composer. The style of a music can be partly characterized by its usage of dissonance, harmony, etc [Gau04]. To measure how much musical style our generations retained from the training pieces, we counted the number of each of the unique 12 harmonic intervals and 12 melodic intervals present in our generated music. A harmonic interval occurs when more than one notes play simultaneously while a melodic interval occurs when the notes are played in a linear order. We also measured 6 normalized characteristics of each of the generated pieces: the percentage of perfect harmonic intervals, the percentage of imperfect consonant harmonic intervals, the percentage of dissonant harmonic intervals, the percentage of perfect melodic intervals, the percentage of imperfect consonant melodic intervals and the percentage of dissonant melodic intervals. We compared our results with the same characteristics of the original musical pieces to evaluate which model captured the most musical style.

Finally, human-composed music generally has some underlying motif or theme that governs the entire piece, a good music generation that mimics human composition should have some similar global structure through out the piece [YM17]. Thus, we have calculated the auto-correlation function, which measures the linear dependence between a observations and past values, of our generated pieces. Music generations that successfully captured the global structure or motif should demonstrate ACF out to more lags.

We expect more restrictive models to generate sequences with less entropy. Due to its Markov property, a simple hmm would allow for more spontaneous generations, and we hypothesize that it would generate music with the highest entropy. HMMs with a more hidden states and/or a dependence structure should have less entropy. We expect deep learning models with long-term memory to have the lowest entropy out of all.

For measuring musical attributes with different intervals, we expect hmms and deep learning models to perform around the same. A music interval is the occurrence of two notes. When they occur simultaneously, they are harmonic, and when they occur sequentially, they are melodic. Although hmms have a much simpler model structure, they do maintain strong pairwise relations. Thus, we would not expect hmms to perform much worse than the deep learning models.

Looking at the model design of our three main models, although HMMs are efficient and robust, music generated by hidden Markov models could potentially have a very limited temporal structure. This is expected because HMMs have very limited memory and are incapable of modeling long term structures. Systems, such as the LSTM, that can retain previous information and understands the underlying global structure would be able to generate a coherent musical piece in line with musical structure. Thus, we hypothesize that LSTM would have higher lags in its ACF than the rest.

4.3 Results

We generated 10 pieces of music from each model, and all of the metric below are the calculated average of the metrics of 10 pieces of music within each model. We also randomly selected 10 Bach chorales and averaged all the metrics as well.

Figure 9 shows the calculated average entropy for pieces generated by different models and the average entropy of original Bach chorales ranked from highest to lowest. Table 1 can be found in Appendix A which lists the average entropy values. From the calculations, we can see that all models generated music with a higher entropy than the original music. This ensures the originality in all

of the music we have generated. The HMMs with different number of hidden states and different level of complex structures all generate music with similar entropy. The LSTM generated music with highest entropy while DMM the lowest.

The harmonic and melodic intervals of the generated music pieces and original Bach pieces are calculated and plotted in Figure 11 and Figure 12, respectively. In these two figures, the music generated by HMMs have similar number of two types of intervals. The music generated from LSTM has the same pattern of harmonic intervals and melodic intervals, which indicates that they resemble the original music style the most. The music generated from DMM has very low interval counts, approximately 0 in each types. The RMSE values between each model and the original are shown in table 3 and table 4.

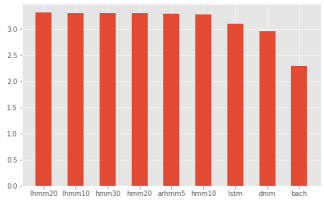

Figure 9: Comparisons

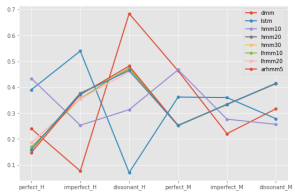

Figure 10: Comparisons between the 6 normalized musical characteristics

The value of the 6 normalized musical characteristics are calculated and shown in Figure 10. The root-mean-square-error was calculated between each model and the original music, and values can be found in table 2. According to the RMSE's, the LSTM model generated music that best retained the 6 musical characteristics from the original Bach pieces, while the DMM retained the least musical style.

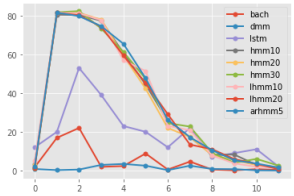

Figure 11: Comparisons of harmonic intervals

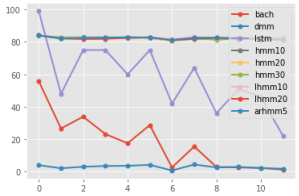

Figure 12: Comparisons of melodic intervals

We have calculated the ACF for all of our models and plotted them against the ACF of original Bach chorales in Figure 13. As we can see from the figure, LSTM preserved the most global structure, and

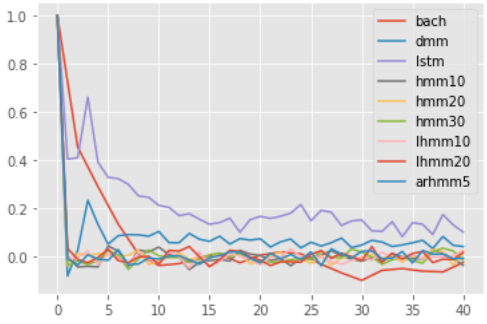

Figure 13: ACF of original Bach and music generated by each model

this is also comfirmed by LSTM's ACF having the lowest RMSE value with the ACF of the original music shown in table 5. The second best model is DMM, who has relatively high acf when lags are small, while all the HMMs generated sequences that almost reveal no temporal dependencies.

4.4 Discussion

Originality: Our hypothesis of model originality was generally proven right. The two deep learning models, being the most restrictive, do have lower entropy than the rest. However, while we expected that the simple HMM would generated the highest entropy among its peers, it is not the case, and what we thought as the most restrictive HMM model, the layered HMM, actually has the highest entropy. When observing the ACF of the HHMs in Figure 13, we see that even if an autoregressive layered HMM was used, the ACF still doesn't show much improvement from the simple HMM. We think that the following might have happened. Although we had hoped that additional hidden states can help increase the temporal structure, due to the nature of the Markov process, the global structure of the model was only improved very slightly, while a lot of entropy was introduced by the additional hidden states that come with the potential for generating more observations.

Musicality: We expected to see similar performance across all models for musicality, however, the LSTM exceeded all of the HMM model, although not by much. The slight under-performance of the HMMs can potentially be caused by the limitation of music annotation. In order for music to be annotated as input for the HMM models, all notes have to be converted into a 1-D linear sequential data. This means that harmonic intervals were annotated as melodic intervals, and the true distribution of harmonic and melodic intervals in the original music were skewed. The under-performance of the HMMs can be caused by training on the skewed data. From Figure 11 and Figure 12, we observe that the music generated form the DMM model hardly had any presence of music intervals. The only reason that DMM had a smaller error on these two metrics in table 3 and table 4 was because they were not normalized, and the counts of intervals in the original Bach pieces happen to be smaller and relatively closer to zero. From table 2, we can see that, surprisingly, DMM performed the worst in terms of retaining musical style from the original pieces. This is unexpected because a complex model with a better ability to capture transitions between hidden states should be able to retain at lease some musicality.

Temporal Structures: As per our expectation, HMM models all performed very poorly when it comes to capturing global structures, and the LSTM model resolved this issue significantly, capturing the global structure of Bach music almost 50% better than all the other models as seen in table 5. Although the global structure of DMM fell far behind the LSTM, it is still better than all the HMM variants, probably due to the fact that complex structure between hidden states are better captured by the DMM. Although not by much, LHMM does seem to capture global structure better than the simple HMM, which is expected given the model design. However, surprisingly, auto-regressive HMM under-performed compared to LHMM in temporal structure, but this can probably be attributed to the fact that due to its complexity, ARHMM was only trained with 5 hidden states, while the LHMM was trained with 20.

Among all the metric and evaluation, the performance within the HMM variants are all pretty similar. It seems that the variations on HMM did not vary in the music generated by much. Among variants, the layered HMM seem to perform the best across all metric.

4.5 Conclusions

The LSTM model ensures originality while scoring the best in retaining musical characteristics and capturing global structure. Overall, it out performed the rest of the models. Besides that, the DMM exceeded the HMM variants in having a temporal structure, while the HMMS performed better at capturing musicality. Taking everything into account, although inefficient and hard to train, deep learning models do out perform simple probabilistic models. As for music generation, models with long term memory structures such as LSTM are generally needed for pieces to have a theme/motif and some music coherency.

For future work, we can investigate more into why the DMM model under-performed so unexpectedly in retaining the musical intervals of different harmonies/melodies from the training music. We should also try different parameters for our deep learning models. Parameter tuning for models such as LSTM are more inefficient due to the fact that the model is very hard to train, and thus takes a long time to give proper feed backs. Perhaps we can look into models with similar memory abilities but a less complex structure. The Gated Recurrent Units (GRU) is a great candidate since it uses a similar gating approach to control information flow, but due to it having only two gates instead of three, it is more computationally efficient and trains faster.

Acknowledgments

We would like to thank Professor Christina Savin for general guidance for the progression of this project and Professor Brain McFee for providing professional advice for choices of music annotation methods.

References

[BBV12] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. *Modeling Temporal Dependencies in High-Dimensional Sequences: Application to Polyphonic Music Generation and Transcription*. 2012. URL: http://www-etud.iro.umontreal.ca/~boulanni/icml2012.

[Cai18] Fangyu Cai. *Google AI Music Project Magenta Drops Beats Like Humans*. 2018. URL: https://medium.com/syncedreview/google-ai-music-project-magenta-drops-beats-like-humans-515de6e5f621.

[Cof92] Don D. Coffman. "Measuring Musical Originality Using Information Theory". In: *Psychology of Music* 20.2 (1992), pp. 154–161. DOI: https://doi.org/10.1177/0305735692202005.

[Cru19] Jeffrey Cruz. *Deep Learning vs Markov Model in Music Generation*. 2019.

[Gau04] Robert Gauldin. *Harmonic Practice in Tonal Music by Robert Gauldin*. W. W. Norton Company, 2004. ISBN: 978-0393152746.

[Keras] *Keras: The Python Deep Learning library*. URL: https://keras.io/.

[KY18] Nikhil Kotecha and Paul Young. *Generating Music using an LSTM Network*. 2018. URL: https://arxiv.org/pdf/1804.07300.pdf.

[KSS16] Rahul G. Krishnan, Uri Shalit, and David Sontag. *Structured Inference Networks for Nonlinear State Space Models*. 2016. arXiv: 1609.09869 [stat.ML].

[Mar97] Yuval Marom. *Improvising Jazz With Markov Chains*. 1997.

[Music21] *music21: a toolkit for computer-aided musicology*. URL: http://web.mit.edu/music21/.

[WG15] Daniel Wysocki and Craig Graci. "Modeling music using hidden Markov models". In: (2015).

[YM17] Anna K. Yanchenko and Sayan Mukherjee. *Classical Music Composition Using State Space Models*. 2017. arXiv: 1708.03822 [cs.SD].

5 Student Contributions

- **Chuan Chen** Music Annotation, LSTM, Hierarchical HMM (We spent much time on implementation of HHMM with no success. Since there are no results for comparison, we disregarded this part in the paper.)
- **Yanqi Xu** Music Annotation, HMM, Two-layered HMM, Autoregressive layered HMM and Deep Markov Model.

Appendix A Tables

Model	Average Entropy
lhmm20	3.322464
lhmm10	3.310403
hmm30	3.308349
hmm20	3.307280
arhmm5	3.292979
hmm10	3.286197
lstm	3.101728
dmm	2.968553
bach	2.291468

Table 1: Average entropy for different models

Model	RMSE of the 6 normalized characteristics
lstm	0.163619
hmm30	0.179776
lhmm10	0.181538
lhmm20	0.183971
hmm20	0.186622
arhmm5	0.187460
hmm10	0.188178
dmm	0.328169

Table 2: RMSE of the 6 normalized musical characteristics between each model and the original music

Model	RMSE of harmonic intervals
dmm	8.109922
lstm	17.494023
lhmm20	39.206728
hmm20	39.398742
hmm10	39.555236
hmm30	39.759118
lhmm10	39.902600
arhmm5	40.084338

Table 3: RMSE of harmonic intervals between each model and the original music

Model	RMSE of melodic intervals
dmm	21.500426
lstm	41.161977
lhmm20	66.266017
hmm20	66.345893
hmm10	66.369603
hmm30	66.539067
arhmm5	66.612993
lhmm10	66.628466

Table 4: RMSE of melodic intervals between each model and the original music

Model	RMSE of ACF
lstm	0.189075
dmm	0.389493
lhmm20	0.404225
lhmm10	0.412734
hmm20	0.416635
arhmm5	0.420553
hmm10	0.425555
hmm30	0.429657

Table 5: RMSE of ACF between each model and the original music

Github: https://github.com/the-yanqi/DS3001-music-generation