EPiC Series in Computing

Volume 77, 2021, Pages 41–51

Proceedings of ISCA 30th International Conference on Software Engineering and Data Engineering

Separated Feature Learning for Music Composition Using Memory-Based Neural Networks

Imad Rahal, Ryan Strelow, Jeremy Iverson, and Katherine Mendel
College of Saint Benedict | Saint John's University, St. Joseph, MN, USA
irahal@csbsju.edu

Abstract

Using algorithms to compose creative and pleasing music has been an ambitious research goal since the 1950s. This trend continues to this day with the help of widely accessible, highly sophisticated music research tools created by big companies, such as Google's *Magenta*. Due to the sequential nature of musical pieces, Recurrent Neural Networks (RNNs), as well as advanced variants such as Long-Short Term Memory networks (LSTMs), have been successfully employed for this purpose. Music scores data is made up of features like duration, pitch, rhythm, chords, etc. As more music features are integrated into the composition process, the space of encodings required to represent possible feature combinations grows significantly, making the process computationally infeasible. This consideration becomes of huge significance in situations with polyphonic pieces, where additional features such as harmonies and multiple voices are present. With an emphasis on efficiency without sacrificing quality, this research aims to further demonstrate the effectiveness of LSTMs for automated music generation by learning from existing music scores data. More specifically, we show that training separated models to learn individual music features and combining results to generate new music is, overall, superior to the common practice of learning resource-intensive complex models that simultaneously incorporate multiple desired features.

1 Introduction

Automated music creation attempts have long failed to stimulate emotions in large part due to lack of creativity and, in many cases, lack of even basic meaningful musical constructs, such as phrasing and motifs. To capture these constructs and contextualize their sequential nature and time-sensitive interdependencies, this research employs specialized artificial neural networks–namely, LSTMs [5], an advanced form of RNNs.

***kern* symbolic music format has thousands of musical pieces meticulously encoded to represent music scores textually. This format is capable of representing nearly every nuance found within a music score, down to the direction the stem of a note is facing on a music sheet. A rich software toolkit called *Humdrum* allows us to extract musical features from ***kern* data. This, in turn, allows us to study the impact on training efficiency and quality of the music pieces generated by combining output from multiple simple models, each focused on a specific music feature, as opposed to employing a single complex model trained over multiple features simultaneously.

1.1 Music Background

The fundamental unit of music is the *note*, with it being composed of *pitch* and *duration* components. Pitch represents the frequency of the sound. Western music theory uses the chromatic scale made up of the following twelve pitch values (also known as semitones or half-steps): *C#* (or *D* ♭), *D*, *D#* (or *E* ♭), *E*, *F*, *F#* (or *G* ♭), *G*, *G#* (or *A* ♭), *A*, *A#* (or *B* ♭), and *B*. A sharp (#) of a pitch raises it by half a step while a flat (♭) lowers it by half. A note can also have no sound at all–often called a *rest*. The *C* major scale is one of the most commonly used concert scales. It contains no sharps or flats making it very convenient to use in music research.

A note duration represents the time value a note is played for. The longest duration is a *whole*, with the remaining duration types commonly being even divisions, like a quarter or an eighth. A duration may be increased through the inclusion of *dots*, such as 50% for a single dot and 75% for a double dot.

The sequential structure of music is key to its appeal. Just as in written language, well thought out order and structure are needed for meaning to be derived. A word on its own is solitary and lacks context; grouping of words follows grammatical rules to produce meaningful phrases and sentences. A sentence is then combined with others to convey a complete thought.

Music is no different in being subjected to necessary structure to form cohesive quality pieces. Structures like *motifs* are found throughout a given composition and can be thought of as a short sequence of notes repeated throughout the piece. Similarly, *phrases* are comprised of multiple motifs. Sequences of interwoven motifs and phrases that comprise a *melody* are musically satisfying making them the core structure that defines a musical piece for many listeners.

Music theory helps promote these structures, and research suggests humans expect the formation of these musical structures in order to engage in a given composition [1][6]. Consequently, the presence of these repeated structures, or lack thereof, carries high importance when attempting to elicit emotions. In order to complete a melody, the beginning of the sequence is usually held in mind while the rest is played–a task carried out by short-term memory; simultaneously, long-term memory maintains the global coherence of the music piece. The existence of both short- and long-term memory is vital for generating melodic and coherent music sequences, thus, motivating the use of LSTMs in this study.

1.2 Related Works

Automated music composition using statistical, data science, or AI techniques is not new, with too many to include in this paper. [4] provides a comprehensive study of techniques employed for this purpose which, among others, include Markov models, rule/constraint/grammar-based approaches, reinforcement learning, and evolutionary algorithms [10].

The sequential structure of music requires the utilization of algorithms capable of detecting spatio-temporal patterns in music pieces in order to learn to compose new ones. Not surprisingly, RNNs have been successfully employed for modeling similar dependencies and structures in a number of applications ranging from text generation and question-answering to building chat bots and sentiment analysis [6]. Furthermore, the use of RNNs, as well as advanced variants like LSTMs, for music

generation has shown great promise [3][7][9]. However, we are not aware of any significant attempts to explore how various ways of incorporating music features into the learning process may impact performance as well as quality, which is the focus of this study.

Moreover, much of the existing work in music composition makes use of MIDI (*Musical Instrument Digital Interface*) or even raw audio data. MIDI is neither a traditional audio format, nor does it represent actual music scores; instead, it describes how notes should be interpreted by a MIDI player. Because of this inherent limitation, many toolkits have been developed to extract music score data from MIDI files, allowing its use in scholarly research; however, such tools tend to be heuristic in nature and, thus, not always accurate [11]. Regardless, as presented in [2], we argue that music composition is about creating music not sound waves. Since musicians typically represent their music symbolically using music scores sheets, we strongly argue that formats like ***kern* are much better suited for this kind of research. ***kern* is text-based and very intuitive to read and edit. Furthermore, the textual nature of ***kern* coupled with the sequential structure of music helps us transform the task of music composition into a sequence manipulation problem.

2 Proposed Approach

Our proposed music generation process is made up of steps, starting with separating and preprocessing of music features of interest from ***kern* data. A number of models are then designed and trained to learn from the extracted feature data. Once the trained models perform satisfactorily, they are used to generate new music.

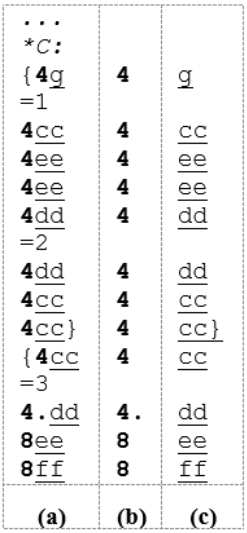
2.1 ***kern* Data & Preprocessing

***kern* is a popular digital music format used to represent music scores textually, making it convenient to view and edit using simple text editors. It encodes vast amounts of information about the music piece in basic structures known as spines or columns. A *spine* contains information to encapsulate one voice in the musical piece–not necessarily limited to human voices and may, instead, represent any musical instrument's part in the piece. Primarily, within each spine are music notes along with measure markers and other musical features found in music sheets. The plethora of commands developed as part of the *Humdrum* toolkit, designed specifically for music research, facilitates extraction and manipulation of desired information from ***kern* files. It also makes it easy to convert to other types of music formats including MIDI [8].

Due to their simplified structure, monophonic music pieces serve as the starting point for this research paving the way for future extensions to consider polyphonic pieces and deal with multiple voices, with chords allowed throughout each voice [11]. Our data corpus contains pieces being either German or Chinese folk music. The choice of genres was largely driven by factors such as data availability and similarities as well as differences between the two–both genres are folk yet quite distinct from each other; regardless, it should be explicitly noted that this should not limit the applicability of findings beyond the two genres.

Our corpus consists of roughly 5k complete samples of music divided equally by genre. All samples have been converted (when needed) into the same *C* major scale and contain the primary voice only, typically the soprano. Figure 1 (a) shows a sample ***kern* file highlighting the duration (b) and pitch (c) values to be extracted from each note.

Extracted duration and pitch feature values for each genre are then used to generate three datasets: two comprised of only the pitch and duration feature values, respectively, and the third combining the extracted pitch and duration values to form complete notes. After further data processing, these datasets are used to train different models allowing us to compare the efficiency and quality of music generated from separated feature models as opposed to a single complex model.



...		
*C:		
{**4**g	**4**	g
=1		
4cc	**4**	cc
4ee	**4**	ee
4ee	**4**	ee
4dd	**4**	dd
=2		
4dd	**4**	dd
4cc	**4**	cc
4cc}	**4**	cc}
{**4**cc	**4**	cc
=3		
4.dd	**4.**	dd
8ee	**8**	ee
8ff	**8**	ff
(a)	**(b)**	**(c)**

Figure 1. (a) Sample ***kern* Data File: (b) Durations (in bold) and (c) Pitches (underlined).

To create the separated duration feature model, duration values are first mapped to the range of 0 and the number of unique duration values (we distinguish between dotted and non-dotted durations). This encoding gives a random yet unique integer "label" for each duration value. The same process is applied to pitch values to generate the separated pitch feature model; for illustrative purposes, a simplified version of the process is depicted in Figure 2 (a) (limited to 8 unique pitch values).

To create the complete note dataset, the encoded pitch and duration values are simply combined to create note encoding labels. It should be clear that the resulting dictionary for possible note labels is significantly larger than the dictionaries used to represent the pitches and durations separately due to the combinatorial nature of this process. To illustrate this by example, prior to preprocessing, German folk samples in our corpus have 30 and 20 pitch and duration labels, respectively, which translates into 600 possible note combinations!

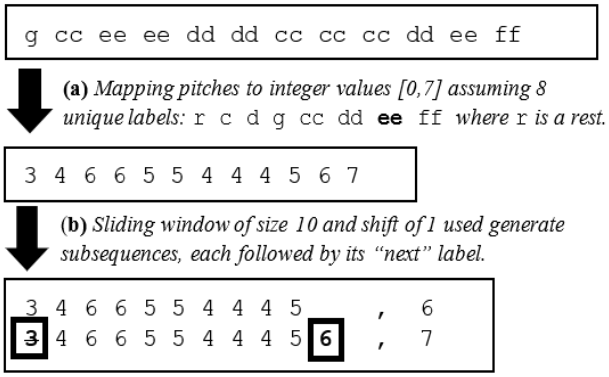


Figure 2. (a) Mapping of Pitches (b) Generating Subsequences.

Now that we have the preprocessed data for all three models, the last step is to extract shorter sequences comprising the final datasets employed for training. Depending on the network used, the models will learn to predict the next pitch/duration/note/etc. label for an input sequence of labels; to facilitate this, we opted to break down each music piece into subsequences using a sliding window of

fixed length to allow the models to look back at a brief "history" of pitch/duration/note labels provided in the input sequence. A sliding window of length 100 was experimentally determined as a good middle ground to balance the desire for coherence and the average length of pieces in the dataset. To generate subsequences of length 100, the sliding window shifts by a single label at a time. Since we are doing supervised learning, we append a class label value to each subsequence, which is simply the actual next label in the piece. The same process (depicted in Figure 2 (b) for a smaller sliding window of size 10) is used to generate input data for the separated pitch, separated duration, and combined note models. The 5k complete music pieces in our corpus generated a total of 147,147 Chinese folk and 276,438 German folk sequences of length 100.

2.2 Network Design

Traditional neural network learning assumes inputs to be largely independent of each other. This makes them inadequate when the training data exhibits spatio-temporal interdependencies where decisions made earlier are expected to factor into future ones. RNNs were first introduced to address this specific issue. They have the same basic structure of a feedforward neural network with the addition of a feedback loop connected to their past decisions to serve as a form of memory. However, RNNs are subject to vanishing and exploding gradients making it hard to "remember" previous decisions especially as the gap between the relevant information and the point where it is needed becomes larger [5].

LSTMs are capable of more accurately learning long-term dependencies. They contain information stored in gated cells, very similar to how computer memory works. The decision on what to keep and when to read and write are controlled via specialized gates. These gates are similar in function to normal neurons in a neural network and work by determining the best action based on signals received. Like other weights in the network, gate weights are adjusted during the learning process to allow LSTMs to learn when specific information needs to be remembered from past events, when newly gained information needs to be stored, or when information can be forgotten altogether due to irrelevance.

There is no definite rule of thumb for how many hidden layers and how many neurons per layer should be used in implementing a network; very often a trial-and-error approach is adequate. We experimentally settled on the following 7-layer network design to facilitate multivariate classification with the exact number of outputs depending on the number of possible labels:

- an input layer containing 100 nodes (equal to the length of the input sequences),
- five hidden layers (more on this later), and
- an output layer containing as many nodes as there are possible label values to predict (uses a *softmax* activation function giving each output node a probability out of 1)

The hidden layers include two LSTM layers, each followed by a dropout layer to help prevent overfitting by ignoring randomly selected neurons during training, and hence reduces the sensitivity to the specific weights of individual neurons. This helps prevent the output from sounding identical to one or more of the original pieces. A dropout rate of 10% was used as a good compromise between retaining model accuracy and preventing overfitting.

2.3 Music Generation Process

As discussed earlier, datasets containing sequences of length 100 are used to train the networks. For a given input sequence, the networks output a list of probabilities, one for each possible label; the label with the highest value is then chosen as the predicted next label.

Music generation is initiated by feeding an existing random seed sequence to a trained model and appending the output label of the model to the end of the input sequence. In order to maintain the same sequence length (of 100, in our case), a label from the beginning of the sequence is then removed to create a new input sequence for the next iteration to "compose" the next part of the sequence, and so on, until complete music pieces are formed, one label at a time.

Figure 3 shows a random pitch sequence being fed into the network as input. The network then generates a probability distribution as output, describing the likelihood of each possible pitch label being next. In this simplified example, there are eight possible pitch labels shown; we would choose label **ee**, as it has a 50% chance of occurring next. This label is appended to the input sequence from the previous iteration and the pitch that is at the front of the sequence is then removed to create a new input sequence for the next iteration. The process continues until sequences of the desired size are created.

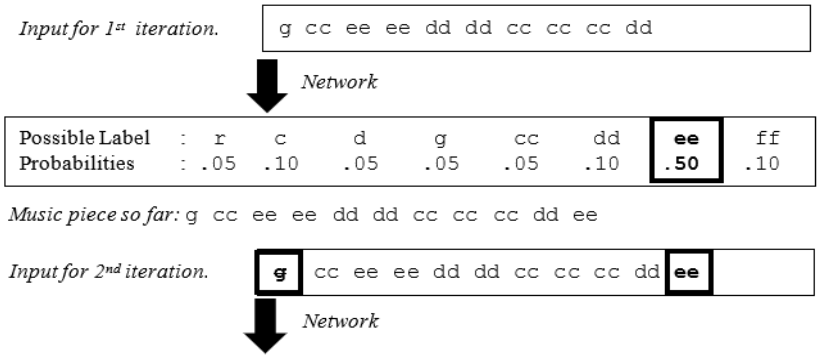


Figure 3. Music Generation.

3 System Design and Experimentation

3.1 Network Training

Three networks are needed for each genre under consideration: one combined note network and two separated pitch and duration feature networks. Results from the separated networks are combined to create complete music notes. Thus, a total of six music generation networks are needed to allow us to consider German and Chinese folk music samples at hand.

	Network	*Label Retained (out of total)*	*Training Data Used*
Chinese	***Complete Note***	113 (296)	62,291
	Duration	19 (30)	80,028
	Pitch	6 (17)	109,288
German	***Complete Note***	121 (338)	42,132
	Duration	18 (30)	86,315
	Pitch	7 (20)	186,688

Figure 4. Training Data.

To expedite and improve the training process, data sequences containing infrequent labels are ignored–recall we produced over 147k Chinese folk and 276k German folk sequences. This results in

smaller datasets that exclude outliers and, more importantly, help distribute the remaining samples per label more evenly. For Chinese folk music, we settled on removing 183 rare occurring notes, resulting in 113 predominant note labels (or 38%). This was chosen experimentally as it performed better than higher or lower percentages in the network training process. To be able to create a similar number of possible notes via the separated pitch and duration networks, only samples with matching pitch and duration labels were retained. A similar process was applied to German folk music; Figure 4 shows the final makeup of the resulting datasets used in this work.

All programs were developed using Google's Tensorflow and executed on a cluster node with the following specifications: Xeon E5-2600 v4 CPU, 20 cores, 32 GB RAM, 12 TB SSD storage. Networks were trained for 30 epochs; GPU acceleration was not used.

To decide on the complexity of the hidden layers, we observed how the number of nodes affected music quality and training time. For each network, we doubled the number of nodes in the hidden layers starting from 32 all the way to 256 nodes per layer. Figure 5 shows the resulting accuracy and training time for each network as a function of the number of nodes in their hidden layers. Accuracy here is defined as the percentage of next labels predicted correctly, averaged over all executions. It is important to highlight the significance of these results given the number of possible labels to choose from; for example, by randomly assigning labels, the accuracy for the Chinese note network would be roughly 0.88%, or 1 out of 113 possible labels as shown in Figure 4–the same applies to all the networks. Consequently, all networks, regardless of complexity, are way better than pure random decisions.

	Network	*32*	*64*	*128*	*256*
Chinese	***Complete Note***	.2085 @ 2,260	.2577 @ 4,340	.3031 @ 5,830	.4544 @ 10,620
	Separate Duration	.6772 @ 2,260	.6857@ 4,820	.6885@ 6,100	.6835@ 9,850
	Separate Pitch	.3944 @ 2,450	.3807 @ 4,830	.4435 @ 5,850	.3501 @ 10,700
German	***Complete Note***	.2174 @ 2,550	.2593 @ 5,150	.3410 @ 7,990	.4992 @ 18,530
	Separate Duration	.7292 @ 3,090	.7464 @ 4,880	.7492 @ 7,860	.7501 @ 17,830
	Separate Pitch	.3940 @ 2,270	.4284@ 3,940	.4675@ 8,000	.4460@ 18,470

Figure 5. Network Design–Column headers show nodes per layer; values show *accuracy @ time (sec)* per 100k samples.

For both genres, we observed the note networks continued to improve in accuracy with added complexity, but the same advances cannot be seen in the pitch and duration networks. This is especially the case for the duration networks for both German and Chinese, where more complex network structure resulted in minimal accuracy gains, if any. We ended up settling on the following: Chinese note network with 256 nodes, Chinese pitch network with 128 nodes, Chinese duration network with 32 nodes, German note network with 256 nodes, German pitch network with 128 nodes, and German duration network with 32 nodes. Final note and pitch networks were chosen as they were the best performers. Because the duration networks performed relatively the same regardless of complexity, the simplest networks were used. The chosen designs are highlighted in Figure 5.

3.2 Objective Evaluation

Along with assessing the ability to create music pieces adhering to different genres, our evaluation aims to highlight differences in efficiency and quality of music generated from complete note networks *vs.* separated feature networks. To this end, we created a new binary classification model

trained to classify German and Chinese folk music. The design for this simple network contains an input layer with as many nodes as the desired length of the generated music pieces, two hidden layers, and an output layer with a single node activated by a binary sigmoid function to output a value close to 0 for German and to 1 for Chinese.

	Accuracy	*Confidence*
Chinese	.9675	.9467
German	.9842	.9734

Figure 6. Evaluation on Real Music.

The evaluation classifier was trained on a set of 100k sequences randomly chosen from the original over 147k Chinese folk and 276k German folk real music sequences (refer to section 2.1), evenly divided between the two genres; all remaining sequences in the original set were used for validation. It is important to highlight that the original data contains labels ignored earlier during the music generation process. This discrepancy is by design to illustrate applicability of results to "real-world" situations.

Figure 6 contains accuracy and confidence results of the trained classifier on the validation set. Confidence here is derived from the 0-1 prediction scale the classifier outputs. Recall that if the network outputs a value in the range 0-0.5, it would be German, and 0.5-1 would be Chinese. Confidence simply records how close we are to 0 for German predictions and to 1 for Chinese ones. Confidence as a whole is computed as the average over all samples used for validation.

To gauge the ability to create music in a specific genre that will satisfy this evaluation classifier, 500 sequences were generated by the trained note network with another 500 sequences created by combining output from the separated pitch and duration networks. The 1k samples for each genre were run through the classifier; the average accuracy and confidence along with the training time needed per 100k samples are reported in Figure 7. Time for separated networks is shown as a range depending on the execution mode–full parallel *vs*. sequential execution.

	Network	*Accuracy*	*Confidence*	*Training Time (sec)**
Chinese	*Complete*	.8520	.82370	10,620
	Combined	.8920	.85140	5,850-8,110
German	*Complete*	.9780	.9504	18,530
	Combined	.9460	.9340	8,000-11,090

Figure 7. Evaluation on Music Generated by Genre using Complete Note vs. Separated Pitch & Duration Networks.

From Figure 7, we can see slight difference in confidence and accuracy values resulting from the approach used to generate music. Considering the Chinese folk results, the separated training of the pitch and duration networks actually shows increased confidence and accuracy. Moving to the German folk results, the separated training was close to meeting the confidence and accuracy of those of the note network. However, a more interesting takeaway from these results is the significantly less time required to generate the music using separated networks!

In other words, while the separated networks for features did not always exceed the average accuracy and confidence values of the note network, they produced very comparable results but with significantly less training time. Time savings anywhere between 23.63% and 44.92% can be observed in the case of Chinese folk music, and 40.15%-56.83% for German folk music, depending on whether the separated networks were trained purely sequentially or completely in parallel.

3.3 Subjective Evaluation

Our assessment of the quality of the generated music pieces so far has relied mostly on "objective" measures. Generally speaking, however, the very nature of art, music included, makes it a creative task, that is almost impossible to assess purely objectively. This paves the way for a more "subjective" evaluation to assess how listeners "feel" about the generated music pieces.

To this end, we created a survey in which listeners were asked to listen to a dozen randomly chosen 30-second generated music samples–6 per genre evenly split between the two compared ways of generating music: 3 complete note, 3 separated feature Chinese folk and the same for German folk. Survey participants were also asked to rate their music background by selecting one of the following levels of expertise that describes them best

(1) No experience or formal education (such as school or college courses)
(2) Limited experience and/or formal education
(3) Extensive experience and/or formal education
(4) Degree or career in music

Level of Expertise	*Count*
Level (1)	7
Level (2)	10
Level (3)	25
Level (4)	8
TOTAL	**50**

Figure 8. Distribution of Survey Participants by Music Background.

The survey presented participants with two original samples of music from the two genres for reference, one Chinese folk and one German folk. Survey participants were asked to listen to these samples as many times as desired to get a good understanding of the characteristics of each genre. The two samples were also made accessible throughout the survey for replay. The survey then presented the 12 generated samples, one at a time in no specific order, and asked participants to categorize each as Chinese or German folk, along with the confidence of their choice. A total of 52 participants took this survey with two eliminated due to being incomplete; the remaining 50 participants are distributed as shown in Figure 8.

Survey results are shown in Figure 9. Although samples used in the survey were randomly chosen from our generated corpus, those generated using separated feature models using our proposed approach were consistently more likely to be identified correctly by survey participants, with improvements ranging approximately from 6% to 10%, depending on which participants are included. The leftmost third part of the chart shows results from all survey participants: the first two bars show results using all 6 samples per approach while the second two bars exclude the sample that confused the largest number of participants (i.e., drop sample with lowest accuracy score); the resulting (rather significant) increase in accuracy illustrates that results were being penalized largely by a few "bad" generated music samples.

We also wanted to better understand how level of expertise impacted the survey results. Hence, the middle third part of the chart ignores results from participants with expertise level (1); similarly, the rightmost third part of the chart includes only results from participants in the top two levels. As expected, participants with more background in music were more likely to correctly classify the test samples. Although results improved with expertise, we note that improvements resulting from using the proposed separated feature approach as opposed to the combined approach did not change much regardless of level of expertise (roughly, 6% using all 6 samples and 10% after the lowest sample is dropped).

A simple explanation for the increase in quality using the proposed separated feature approach could be attributed to the fact that, when generating music by learning features separately before combining results, we are more likely to generate a larger set of possible combinations compared to the combined features approach (which is limited to existing combinations). This, of course, remains open for further exploration.

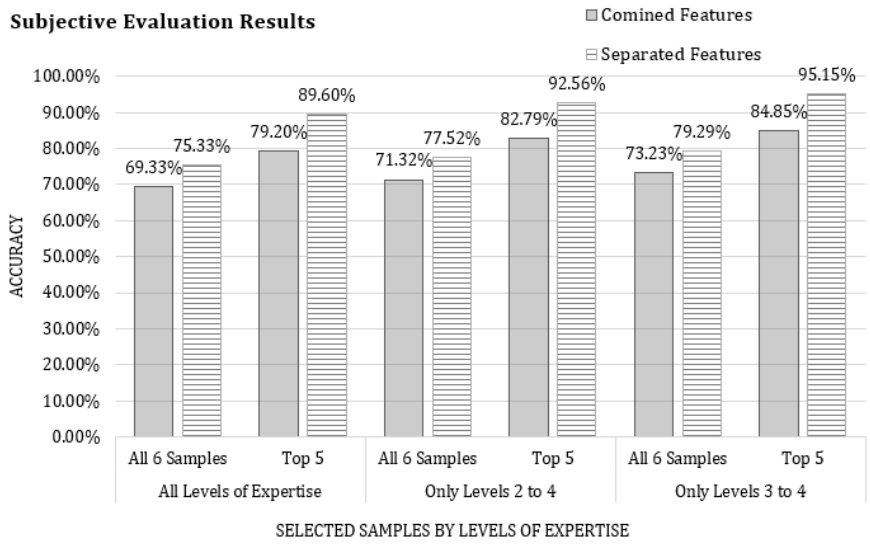


Figure 9. Survey Results.

4 Conclusion and Future Work

Researchers are still only scratching the surface of what is possible with the use of computers in areas traditionally reserved for purely artistic creation, such as music composition. This research demonstrates the potential benefits of feature separation in music composition from music scores data. Although the idea is simple, experimental results show that music generation by combining results from multiple simple models requires significantly less training time whilst maintaining quality steady (if not improving it) compared to using a single more complex model. These results are further supported by a survey clearly highlighting that participants are significantly more likely to correctly identify music generated using the proposed separated features approach, which dramatically increases our confidence in this direction. Further analysis to understand the reasoning behind this increase in quality is currently ongoing.

These advantages may seem novel when creating simple melodies, but the true potential lies in polyphonic music, where complexity and overall time efficiency increase drastically along with the number of musical features. Consequently, we plan to extend this work beyond two features especially for polyphonic music data.

We also plan to explore the use of genetic algorithms to evolve the generated music over time thus improving quality. This requires us to devise suitable fitness functions to determine what solutions are deemed “good”. The challenge here, however, is that in music, “good” is subjective and highly dependent on context. We must also consider how to capture creativity in fitness functions.

References

[1] Bevington, J., and Knox, D. 2014. Cognitive Factors in Generative Music Systems. In Proceedings of the ACM International Audio Mostly Conference: A Conference on Interaction with Sound, 1-8. Aalborg, Denmark: ACM Press.

[2] Briot, J.P., and Pachet, F. 2020. Deep Learning for Music Genera-tion: Challenges and Directions. Neural Computing & Applications 32: 981-993.

[3] Gunawan, A.A.S.; Iman, A.P.; and Suhartono, D. 2020. Automat-ic Music Generator Using Recurrent Neural Network. Internation-al Journal of Computational Intelligence Systems 13(1): 645-654.

[4] Herremans, D.; Chuan, C.H.; and Chew, E. 2017. A Functional Taxonomy of Music Generation Systems. ACM Computing Surveys 50(5).

[5] Hochreiter, S., and Schmidhuber, J. 1997. Long Short-term Memory. Neural Computation 9(8): 1735-1780.

[6] Ivanov, I; Jing, L.; and Dangovski, R 2018. Improving the Performance of Unitary Recurrent Neural Networks and Their Application in Real-life Tasks. In *Proceedings of the International Conference on Computer Systems and Technologies*, 6-11. Ruse, Bulgaria: ACM Press.

[7] Papakostas, M.K.; Gkiokas, A.; and Katsouros, V. 2018. Interactive Control of Explicit Musical Features in Generative LSTM-based Systems. In *Proceedings of the ACM International Audio Mostly Conference on Sound in Immersion and Emotion*, 1-7. Wrexham, UK: ACM Press.

[8] Sapp, C.S. 2005. Online Database of Scores in the Humdrum File Format. In *Proceedings of the International Conference on Music Information Retrieval*, 664-665. London, UK.

[9] Shah, F.; Naik, T.; and Vyas, N. 2019. LSTM Based Music Generation. In Proceedings of the IEEE International Conference on Machine Learning and Data Engineering, 48-53. Taipei, Taiwan: IEEE.

[10] Shi, N., and Wang, Y. 2020. Symmetry in Computer-Aided Music Composition System with Social Network Analysis and Artificial Neural Network Methods. *Journal of Ambient Intelligence and Humanized Computing*: 1-16.

[11] Thickstun, J.; Harchaoui, Z.; Foster, D.; and Kakade, S. 2019. Coupled Recurrent Models for Polyphonic Music Composition. In Proceedings of the International Conference on Music Information Retrieval, 311-318. Delft, The Netherlands.