Combining LSTM and Feed Forward Neural Networks for Conditional Rhythm Composition

Dimos Makris[1], Maximos Kaliakatsos-Papakostas[2], Ioannis Karydis[1], and Katia Lida Kermanidis[1]

[1]Department of Informatics, Ionian University, Corfu, Greece,
{c12makr,karydis,kerman}@ionio.gr
[2]Institute for Language and Speech Processing, R.C. "Athena", Athens, Greece,
maximos@ilsp.gr

Abstract. Algorithmic music composition has long been in the spotlight of music information research and Long Short-Term Memory (LSTM) neural networks have been extensively used for this task. However, despite LSTM networks having proven useful in learning sequences, no methodology has been proposed for learning sequences conditional to constraints, such as given metrical structure or a given bass line. In this paper we examine the task of conditional rhythm generation of drum sequences with Neural Networks. The proposed network architecture is a combination of LSTM and feed forward (conditional) layers capable of learning long drum sequences, under constraints imposed by metrical rhythm information and a given bass sequence. The results indicate that the role of the conditional layer in the proposed architecture is crucial for creating diverse drum sequences under conditions concerning given metrical information and bass lines.

Keywords: LSTM, neural networks, deep learning, rhythm composition, music information research

1 Introduction

Attempting to imitate human creativity has long been an interesting direction for computer scientists. Music has received significant attention in relation to other arts such as graphical arts, painting, dance, or architecture due to its rigorous formalisation, available from the early stages of its evolution [19]. The work of Hiller and Isaacson [7], on the composition of a musical piece using a computer program, was published as early as shortly after the introduction of the very first computer.

Among numerous definitions of Algorithmic (or Automatic) Music Composition (AMC) [23,10], D. Cope[1] provided an interesting alternative, "a sequence (set) of rules (instructions, operations) for solving (accomplishing) a [particular] problem (task) [in a finite number of steps] of combining musical parts (things, elements) into a whole (composition)" [20].

AMC is a complex problem with tasks ranging from melody and chords' composition to rhythm and lyrics, among others [2]. Its applications include

[1] Panel Discussion in the ICMC '93.

varying degrees of combination of computational creation (e.g. fully automated background/harmonisation music or assisted co-composition) with humans' creativity for the production of musical works. AMC has been approached with a plethora of methods from various points of view.

Artificial Neural Networks (ANNs) acted as an extra powerful computation tool to extend the available methods which are usually probabilistic models. A number of research works have been published using ANNs and, especially lately, Deep Learning architectures for composing music (e.g. [6,13]) and most of these are using LSTM layers.

Recurrent neural networks, especially LSTM networks, have been utilized for music generation (see [22] for further references), since they are capable of modeling sequences of events. However, hitherto proposed architectures focus on the generation of sequences *per se*, without considering constraints. For instance, the performance of human drummers is potentially influenced by what the bass player plays. Additionally, human drummers could play drums on a time signature they have never played before, e.g. 15/8, by utilising the knowledge they have obtained only by practicing (learning) 4/4 beats; this happens because human drummers have an *a priori* understanding of metric information which helps them perform drum rhythms on meters they have never seen before. In this work, we propose the combination of different neural network layers, i.e. feedforward and recurrent, for composing drum sequences based on external information, i.e. metric and bass information.

The remainder of this paper is organised as follows: Section 2 presents background information and existing research on Automated Musical Composition and related notions. Next, Section 3 presents the proposed combination of LSTM and Feed Forward Neural Networks for conditional rhythm composition. Section 4 details the experimental evaluation of the proposed method, while the work is concluded in Section 5.

2 Related Work

Algorithmic Musical Composition refers to the the type of creativity that does not focus on "flash out of the blue", as would be the result of inspiration or genius, but a process of incremental and iterative revision, more similar to hard work [10] in the post-digital computer age. Artificial neural networks (ANNs) have long been utilised for the purposes of automated composition[2]. Their extended use in AMC is due on the capability of ANNs to resemble human creative activities despite the expensive training required in order to do so.

A very common version of ANNs is the feedforward ANN, which include neurons/units that are commonly divided in three types of layers: input, hidden, and output. Units between different layers are interconnected by weights that are multiplied with the values of their respective input unit. The resulting output is then propagated to a transfer function through units to the output unit. The "learning" procedure of feedforward ANNs refers to their ability to modify the

[2] Papadopoulos & Wiggins [20] compiled an extensive such list, dating back to 1992.

connecting weights for producing output that optimally matches known target values (supervised learning), by minimising the error within a threshold. Recurrent Neural Networks (RNNs) on the other hand, have a similar architecture but the hidden layers are more sophisticated, including recurrent connections for remembering past events. RNNs have been extensively utilised for AMC purposes such as generation of chord sequences [15] and melodies [18].

Long Short-Term Memory (LSTM) is a form of RNN initially proposed by Hochreiter and Schmidhuber [8]. Following the principles of RNNs, LSTM networks are generic in the sense that given adequate network units, LSTMs allow for any conventional computation. In contrast to RNNs, LSTMs are more suited to learning from experience to classify, as well as to processing and predicting time-series, when there are time lags of unknown size and bound between important events. LSTMs feature relative insensitivity to gap length, thus being advantageous to alternative forms of RNNs or hidden Markov models. Accordingly, LSTMs have been utilised for AMC purposes such as learning chord progressions [5], drum progressions [2] and learning generic percussion tracks [14] as well as creating musically-appealing four-part chorales in the style of Bach [6] and folk tunes [22].

One of the key advantages of applying ANNs for ACM is the lack of requirement for a priori knowledge, such as rules, constraints, of the domain since ANNs learn through the examination of input examples. Nevertheless, ANNs' application in AMC is not without drawbacks, namely the diminished capability of mapping input sequences to output higher-level features of music and the difficulty of generalisation related to the input examples' size [20].

3 Conditional Rhythm Composition with LSTM and Feed Forward Neural Networks

In this paper we introduce an innovative architecture for creating drum sequences by taking into account the drum generation of previous time steps along with the current metrical information and the bass voice leading. Section 3.1 describes the collection and preprocessing methodology for the training data, while Section 3.2 presents the model's architecture.

3.1 Data Representation

The utilised corpus consists of 45 drum and bass patterns with 16 bars each, in 4/4 time signature from three different rock bands. All the pieces were collected manually from web tablature learning sources [3] and then converted to MIDI files by keeping the drum and bass tracks only. For each track, we selected a characteristic 16 bar snippet which is usually detected on the chorus part of a rock song. The selection was done with the help of a student of the Music Department of the Ionian University, Greece.

We used two different input spaces to represent the training data and feed them to two different two different types of ANNs, that are afterwards merged

[3] `http://www.911tabs.com/`

into a common hidden layer. The first one, an LSTM network, corresponds to the drum representation while the second one, the feedforward network, represents information of the bass movement, and the metrical structure information.

As far as the drums were concerned, their representation was based on text words, as proposed by [2] with the following alterations. To encode simultaneous events in a track into texts, we used the binary representation of pitches, i.e., standard components of drums - kick, snare, hi-hats, cymbals and toms. We also limited the number of events in a bar to 16 by quantising every track to the closest 16th-note. This process lead to 256 word music events for each learning input. Due to the limited training data, and for efficient representation and learning, only five components were retained; kick, snare, any tom event, open or closed hi-hats, and crash or ride cymbals. For example, 10010 and 01010 represents a time step with simultaneous playing of kick and hi-hat followed by simultaneous playing of snare and hi-hat. There can be theoretically $2^5 = 128$ words, but there are indeed much fewer, since the combinations of drum components that are actually played are limited.

Moving on to the bass, we use information regarding the voice leading (VL) of bass (which has proven a valuable aspect in harmonisation systems [17], [9]). Specifically, VL was defined by calculating the pitch difference of the bass between two successive time steps, representing this information in a 1x4 binary vector. The first digit of this vector declares the existence of a bass or rest event, while the three remaining digits show the calculation of the bass voice leading in the following 3 different cases: [000] steady VL, [010] upward VL and [001] downward VL. For example, if the bass pitch from 42 changes successively to 35, then to 40 and finally to a rest, the Bass VL vectors occurring are [1001], [1010] and [0000].

In addition to the bass information we included a 1x3 binary vector representing metrical information for each time step. This information ensures that the network is aware of the beat structure at any given point. Considering the simple beat structure (resolution of 16ths and 4/4 time signature) of the examined examples, the first digit declares the start of a bar, the 2nd of half-bar, and finally the 3rd of quarter bar. So for example the first 9 time steps of a sequence could be translated to metrical information as: [111], [000], [000], [000], [001], [000], [000], [000], [011]. This representation can be arbitrarily expanded or modified to reflect additional or other information. For instance, a 4th digit can be included that indicates, e.g., the beginning of the chorus. Additionally, the second and third digits can be used for representing beat accents rather than beat positions. For instants, different binary representations can be used for denoting two different accentuation patterns in 7/8 time signature, e.g., 4-3 and 3-2-2. The examination of such possibilities is left for work.

3.2 Proposed Architecture

The proposed architecture consists of 2 separate modules for predicting the next drum event. An LSTM module learns sequences of consecutive drum events, while a feedforward layer takes information on the metrical structure and bass movement. The output of the network is the prediction of the next drum event. We used the Theano [1] deep learning framework and Lasagne [3] library. Figure 1 shows a diagram of our proposed architecture.

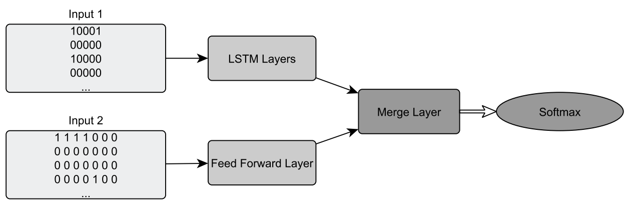


Fig. 1: Proposed Deep Neural Network Architecture for Conditional Drum Generation.

For the drum input space we used 2 stacked LSTM layers with 128 or 512 Hidden Units and a dropout of 0.2, similarly to [25]. The LSTM layers have 16 time steps memory, which correspond to a full bar. Accordingly, the LSTM attempts to predict the next step in a stochastic manner. In each prediction for time index n, the network outputs the probabilities of every state. The bass and metrical input space is then fed into the Hidden module on a dense fully connected layer. Subsequently, a merge layer is used in order to take the output of each module that is then passed through the *softmax* nonlinearity to generate the probability distribution of the prediction. During the optimisation phase, the AdaGrad loss function [4] is calculated as the mean of the (categorical) cross-entropy between the prediction and the target.

4 Results

The LSTM methodology has been used for generating sequences that reflect a learned style, specifically using similar representations of rhythms as the one followed herein. Since the aim of the conditional rhythm composition methodology is to employ conditional information in a neural network fashion, our experimentation focuses on the following questions:

1. Are the training and generation processes negatively affected by the introduction of the feed forward (conditional) layer?
2. Does the feed forward (conditional) layer play any role in the rhythms generated by the system and if so, in what aspects?
3. How are the capabilities of the system affected when trained on datasets with different characteristics?

To answer the aforementioned questions, simple examples of generated rhythms were analysed, allowing the exploration of conditional layer's impact. Then, two larger-scale experiments are conducted where the features of composed and "ground-truth" rhythms were compared, focusing on the role of the conditional layer and the characteristics of different training scenarios, respectively. Those experiments allowed a deeper view into what the proposed methodology achieves and revealed the main weakness of related methodologies (including the proposed one): their incapability to capture high-level structure; suggestions for future enhancements are proposed in the concluding section of the paper (Section 5).

In the following examples and experiments, 5 different networks were trained:

1. PTNN: trained with 15 excerpts from the Porcupine Tree band.
2. FloydNN: trained with 15 excerpts from the Pink Floyd band.
3. QueenNN: trained with 15 excerpts from the Queen band.
4. allNN15: trained with 15 excerpts in total – 5 randomly selected from each aforementioned band.
5. allNN: trained with all 45 excerpts of the aforementioned bands.

Especially for the experiments in Section 4.2, where the role of the conditional layer was examined, a custom version of those networks is tested that does not include the conditional layer, i.e. those networks were trained and generated as typical LSTM networks without the conditional parts. Those networks are marked with a "no" suffix, e.g. the "PTNN" without the conditional layer is named as "PTNNno".

4.1 Examples of Generated Rhythms

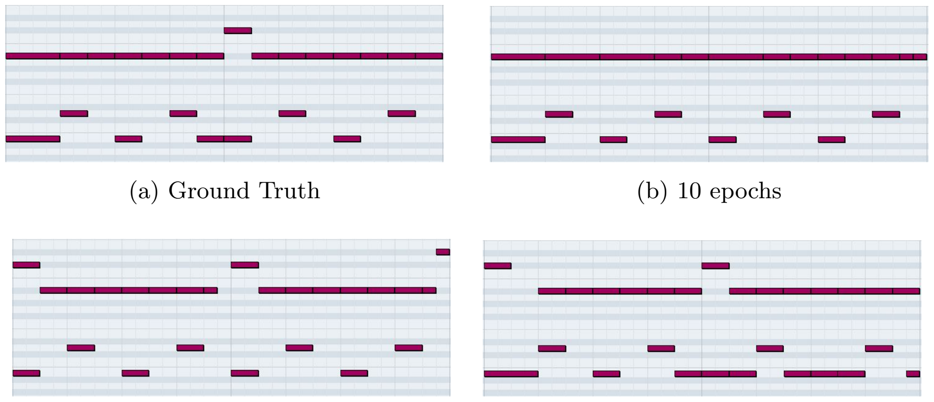
Figure 2 shows three different examples of drum generation of the *allNN* model of two bars from a song of Queen which was not included on the training dataset. Starting from bottom to top in the piano-roll drum depiction, we identify in the following order kick, snare, toms, hi-hat and crash/ride events. According to the Ground Truth of the example, the rhythm is very simple with a standard pattern on kick snare and hi-hat events and a crash in the end of the first bar. In addition the bass harmony is also simple with few onsets and large durations.

As shown in Figure 2's epoch progressions, the network is adjusted from the early stages of learning. On *epoch*10 it is able to understand the metric structure of the song and create simple and repeating drum patterns. Moving forward on the training the network gives more complex generations. On *epoch*30 the generation is almost the same as the original with minor difference in some additional cymbals (crash-ride) on the start and at end of the excerpt. At this stage, the Condition Layer seems to contribute a lot on the generation, as evident in the last example.

Then, we edited the bass track and replaced it with a complex playing with several more onsets in total. The Conditional Layer acted accordingly, forcing the network to create drum events which can follow this particular bass harmony. This is evident in Figure 2 (d) by the additional kick events created in both bars.

4.2 Training with vs. without the Conditional Layer

The characteristics of drums' rhythms can be examined by extracting qualitative features, similarly to the ones presented in [11]. However, in order to extract such


(a) Ground Truth (b) 10 epochs

(c) 30 epochs (d) 30 epochs with edited Bass

Fig. 2: Piano Roll drum generation samples of the proposed network.

features the representation of rhythms requires transformation in a compatible representation, where only three basic drum elements are included: hi-hat, snare and kick drum (`H`, `S` and `K`). Since the current representation incorporates 5 elements, the snare and tom elements (rows) are merged into a single snare row, the cymbals rows (hi-hat and crash) are mapped into a single hi-hat row while the kick row remains as is.

Following the methodology found in [11], 32 features were extracted (described in Table 1), forming a vector representing each rhythm in the database, $\boldsymbol{r_i} = \{f_1^{(i)},\ f_2^{(i)},\ \ldots,\ f_{32}^{(i)}\}$, where i is the index of the rhythm; those features incorporate the concepts of "density", "syncopation" [21], "symmetry" [12] and the "weak-to-strong" [11] ratios (measuring the ratios of the total intensities of weak over the total intensities of distinctive events), in the distinctive beats of the rhythm and in each separate percussive element (features 1-16). The distinctive beats of a rhythm are considered to be the beats in the snare or the kick drum that exceed the 70% of the maximum intensity in this rhythm. Since rhythms in this work are assumed binary, the distinctive beat simply describes the locations of snare and/or kick, expressed as a 1-row binary array. Features 17-19 and features 24-26 capture simultaneous pairs and isolated onsets respectively, while features 20-23 describe transition probabilities between the `S` and `K` elements. The mean values and standard deviations of the intensities of each percussive element are described in features 27-32. Since the examined rhythms are excerpts of 16 bars, each rhythm is represented by an array of $32 \cdot 16 = 512$ values.

In order to examine the efficiency of the introduced conditional layer, 6 "ground-truth" pieces were used – two from each band (Porcupine Tree, Pink Floyd and Queen) – none of which was included in the training sets. Both the simple LSTM and the conditional versions of all networks generated drum rhythms with initial seeds given by the ground-truth pieces, while the conditional net-

Table 1: The employed drums' features.

Feature indexes	Feature description
1–4	density, syncopation, symmetry and weak-to-strong ratio of the distinctive beat
5–16	density, syncopation, symmetry and weak-to-strong ratio of each drum element
17–19	percentages of simultaneous pairs of drums onsets (`H`–`K`, `H`–`S` and `S`–`K`)
20–23	percentages of transitions between all combinations of `K` and `S`
24–26	percentages of isolated `H`, `S` or `K` onsets
27–32	intensity mean value and standard deviation for each drum element

works were also given the metrical and bass voice information of the ground-truth. The compositions of each networks after every 5 epochs of training were extracted until epoch 30. The features of every composition were then extracted and mapped, along with the features of the ground truth rhythms, in two dimensions using the t-SNE [24] technique.

Figure 3 illustrates the 2D mapping of all features (ground-truth and compositions in all epochs by all network versions) with colors according to the 5 clusters assigned by k-means clustering [16]; this number of clusters was used partially because of the fact that 5 different networks were trained and partially because of the clarity of the provided results after experimentation with different cluster numbers. The assessment of the characteristics of each cluster is performed by counting the occurrences of different training "keywords" in each cluster, i.e., the trained network name and the training epochs that composed a rhythm.

The aggregated keywords in each cluster are shown in Table 2. The cluster of main interest is C_3 since this is the cluster where the ground truth rhythms are. It is evident from the third column in Table 2 that the *allNN* network was mostly able to capture the characteristics of the ground truth rhythms. The pure LSTM rhythms (triangles), composed without using the conditional layer, are mainly placed far away from the ground-truth cluster. This fact indicates that the conditional layer does indeed contribute in the production of rhythms that more accurately resemble the characteristics of the trained style; at the same time, those rhythms have relatively diverse characteristics, which is evident by the dispersion in C_3. Therefore, the use of the conditional layer is shown to improve the efficiency of the network in capturing the characteristics of the training rhythms, while the diversity of the produced rhythms is preserved.

4.3 Examining the role of the conditional layer

To examine the role of the conditional layer, a new version of the 3 out of the 6 ground truth rhythms was produced by the first author of this work, with the bass being modified. The modification in all 3 cases was mild and

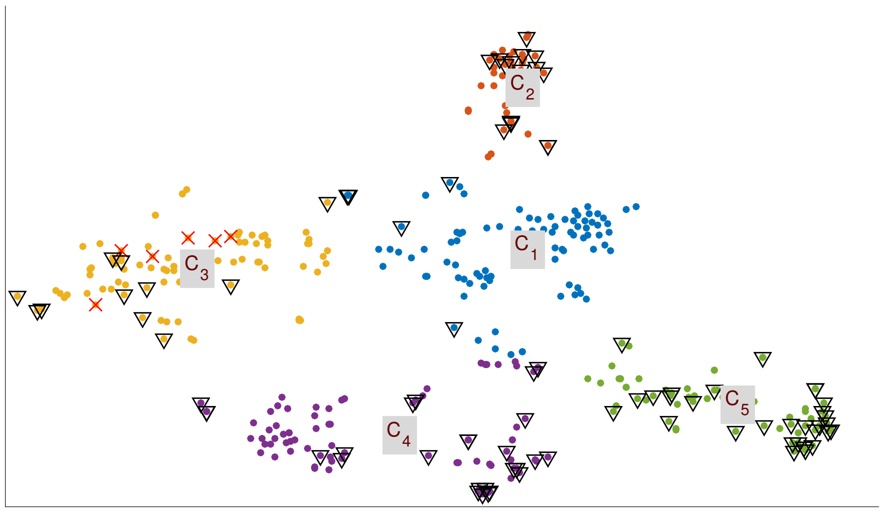


Fig. 3: Two-dimensional mapping of the features of all network compositions with and without (circumscribed by triangles) the conditional layer. The ground-truth features mapping is illustrated with "×"s. Coloring is based on k-means clustering with 5 clusters.

Table 2: The six most common labels in each cluster in Figure 2 along with the numbers of their occurrences.

C_1	C_2	C_3	C_4	C_5
QueenNN: 29	epoch5: 13	allNN: 24	epoch30: 21	allNNno: 18
epoch5: 19	epoch10: 13	epoch20: 17	epoch20: 19	FloydNNno: 18
allNN15: 18	FloydNNno: 13	epoch25: 16	epoch15: 17	epoch25 17
epoch10: 17	allNN15no: 11	epoch10: 15	PTNN: 16	PTNNno: 15
FloydNN: 16	QueenNNno: 9	epoch5: 14	FloydNN: 15	epoch30: 13
epoch25: 15	epoch30: 4	epoch15: 13	PTNNno: 14	epoch20: 11

compatible with the style/original version of each piece. The modifications in the bass were expected to affect the system when composing, since the conditional layer changes. The influence of the changes in the bass are illustrated in Figure 4, where the features of all 16 bars of the rhythms were reduced to 2 dimensions. The coloring is again the result of clustering with the k-means with 5 clusters, while the rhythms in circles are the ones composed with the use of the modified bass in the conditional layer.

The most common keywords in each cluster (composing network and epoch of training) are show in Table 3. The cluster of interest is C_3 where the ground truth rhythms are included. Again the *allNN* has the strongest presence in this cluster, a fact that, combined with the findings in Table 2, indicates that more training data (*allNN* is trained with 3 times as much data as all other networks) lead to

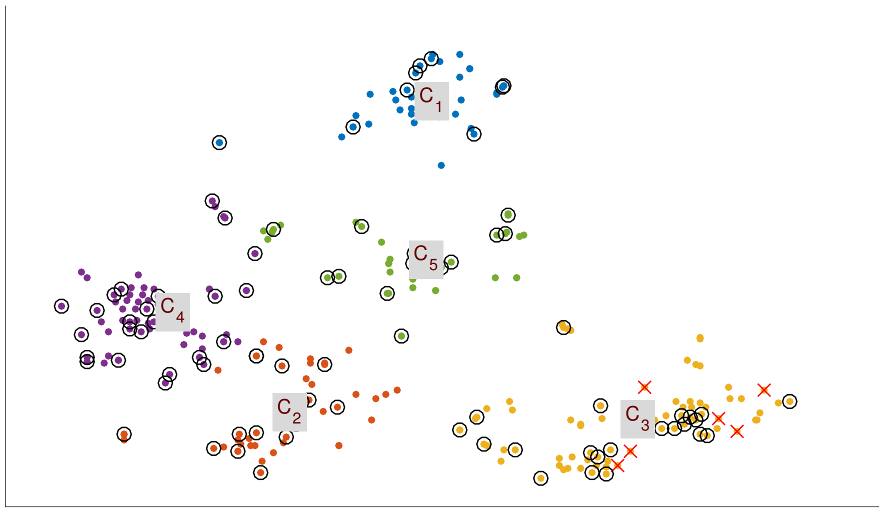


Fig. 4: Two-dimensional mapping of the features of all network compositions with original and modified bass (circumscribed by circles). The ground-truth features mapping is illustrated with "×"s. Coloring is based on k-means clustering with 5 clustes.

the generation of rhythms that are closer to "real rhythms" of the learned style. Additionally, the presence of circled rhythms (composed with modified bass in the conditional layer) within this cluster indicates that the proposed system does not get "lost" when presented alternative scenarios and continues to compose rhythms that are close to "real rhythms" (ground truth). It should be noted that the evaluation of the rhythms composed with the modified bass are different from the ones composed with the original bass – i.e. the circled rhythms are not on top of non-circled rhythms. Therefore, the modifications in the bass potentially affect the network, but in a way that appears to be meaningful.

Table 3: The six most common labels in each cluster in Figure 3 along with the numbers of their occurrences.

C_1	C_2	C_3	C_4	C_5
FloydNN: 19	allNN15: 19	allNN: 34	QueenNN: 30	allNN15: 16
PTNN: 16	QueenNN: 12	PTNN: 20	FloydNN: 19	epoch10: 11
epoch30 13	epoch20: 9	epoch25: 16	epoch5: 15	PTNN: 11
epoch20: 12	epoch10: 9	epoch5: 14	epoch30: 11	epoch15: 7
epoch15: 12	FloydNN: 9	epoch10: 14	epoch25: 11	epoch5: 6
allNN: 8	epoch5: 8	epoch20: 12	epoch10: 10	epoch30: 6

5 Conclusions

In this work, we introduced an innovative architecture approach for creating drum sequences. The proposed Neural Network consists of a Recurrent module with LSTM layers which learns sequences of consecutive drum events and a Feed Forward Layer which takes information on the metrical structure and the bass movement.

Despite the few training data, the experiments showed promising results highlighting the importance of the Feed Forward Layer. LSTM systems without the conditional layer learn and compose music regardless of given conditions. Therefore, a drumming system that is based on a simple LSTM architecture does not get affected by, e.g., potential changes in the bass, which is not the case with human drummers.

Additionally, the preservation of a metrical structure in simple LSTM systems is only dependent on their ability to learn the metric structure these are trained on. The conditional layer enables the LSTM networks to simulate humans in both tasks: respond to changes in other instruments (e.g. bass) and "tune-in" to certain metrical structures.

Our future work will include the extension of the available training data, along with the expansion of network architecture with additional Feed Forward Layers featuring other rhythm instruments (e.g. guitar) or rhythm information. Finally more experiments will be conducted to test the system's behavior in metrical structures not trained on (e.g. 15/8) as well as how the accentuation patterns can be represented and learned.

References

1. James Bergstra, Frédéric Bastien, Olivier Breuleux, Pascal Lamblin, Razvan Pascanu, Olivier Delalleau, Guillaume Desjardins, David Warde-Farley, Ian Goodfellow, Arnaud Bergeron, et al. Theano: Deep learning on gpus with python. In *NIPS 2011, BigLearning Workshop, Granada, Spain*, volume 3. Citeseer, 2011.
2. Keunwoo Choi, George Fazekas, and Mark Sandler. Text-based lstm networks for automatic music composition. *arXiv preprint arXiv:1604.05358*, 2016.
3. Sander Dieleman, Jan Schlüter, Colin Raffel, Eben Olson, Søren Kaae Sønderby, Daniel Nouri, Daniel Maturana, Martin Thoma, Eric Battenberg, Jack Kelly, et al. Lasagne: First release. *Zenodo: Geneva, Switzerland*, 2015.
4. John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.
5. Douglas Eck and Juergen Schmidhuber. A first look at music composition using lstm recurrent neural networks. *Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale*, 103, 2002.
6. Gaëtan Hadjeres and François Pachet. Deepbach: a steerable model for bach chorales generation. *arXiv preprint arXiv:1612.01010*, 2016.
7. Lejaren Arthur Hiller and Leonard M Isaacson. *Experimental Music; Composition with an electronic computer.* Greenwood Publishing Group Inc., 1979.

8. Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
9. Dominik Hörnel. Chordnet: Learning and producing voice leading with neural networks and dynamic programming. *Journal of New Music Research*, 33(4):387–397, 2004.
10. Bruce L Jacob. Algorithmic composition as a model of creativity. *Organised Sound*, 1(03):157–165, 1996.
11. M. Kaliakatsos-Papakostas, A. Floros, and M. N. Vrahatis. Evodrummer: deriving rhythmic patterns through interactive genetic algorithms. In *International Conference on Evolutionary and Biologically Inspired Music and Art*, pages 25–36. Springer, 2013.
12. M. A. Kaliakatsos-Papakostas, A. Floros, M. N. Vrahatis, and N. Kanellopoulos. Genetic evolution of L and FL–systems for the production of rhythmic sequences. In *In Proceedings of the 2nd Workshop in Evolutionary Music (GECCO 2012)*, pages 461–468, Philadelphia, USA, 7-11 July 2012.
13. Vasanth Kalingeri and Srikanth Grandhe. Music generation with deep learning. *arXiv preprint arXiv:1612.04928*, 2016.
14. A Lambert, Tillman Weyde, and Newton Armstrong. Perceiving and predicting expressive rhythm with recurrent neural networks. In *Proceedings of the 12th International Conference in Sound and Music Computing, SMC 2015*. SMC15, 2015.
15. JP Lewis. *Algorithms for music composition by neural nets: improved CBR paradigms*. Ann Arbor, MI: Michigan Publishing, University of Michigan Library, 1989.
16. James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA., 1967.
17. Dimos Makris, Maximos A Kaliakatsos-Papakostas, and Emilios Cambouropoulos. Probabilistic modular bass voice leading in melodic harmonisation. In *ISMIR*, pages 323–329, 2015.
18. Michael C Mozer. Neural network music composition by prediction: Exploring the benefits of psychoacoustic constraints and multiscale processing. *Musical Networks: Parallel Distributed Perception and Performance*, 227, 1999.
19. François Pachet and Pierre Roy. Musical harmonization with constraints: A survey. *Constraints*, 6(1):7–19, 2001.
20. George Papadopoulos and Geraint Wiggins. Ai methods for algorithmic composition: A survey, a critical view and future prospects. In *AISB Symposium on Musical Creativity*, pages 110–117. Edinburgh, UK, 1999.
21. G. Sioros and C. Guedes. Complexity driven recombination of midi loops. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR)*, pages 381–386, Miami, USA, October 2011. University of Miami.
22. Bob Sturm, João Felipe Santos, and Iryna Korshunova. Folk music style modelling by recurrent neural networks with long short term memory units. In *16th International Society for Music Information Retrieval Conference (ISMIR)*, 2015.
23. Martin Supper. A few remarks on algorithmic composition. *Computer Music Journal*, 25(1):48–53, 2001.
24. Laurens van der Maaten. Learning a parametric embedding by preserving local structure. *RBM*, 500(500):26, 2009.
25. Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014.