MINGUS: MELODIC IMPROVISATION NEURAL GENERATOR USING SEQ2SEQ

First Author
Affiliation1
author1@ismir.edu

Second Author
Retain these fake authors in submission to preserve the formatting

Third Author
Affiliation3
author3@ismir.edu

ABSTRACT

Sequence to Sequence (Seq2Seq) approaches have shown good performances in automatic music generation. We introduce MINGUS, a Transformer-based Seq2Seq architecture for modelling and generating monophonic jazz melodic lines. MINGUS relies on two dedicated embedding models (respectively for pitch and duration) and exploits in prediction features such as chords (current and following), bass line, position inside the measure. The obtained results are comparable with the state of the art, with particularly good performances on jazz music.

1. INTRODUCTION

Natural Language Processing (NLP) techniques are achieving remarkable results when applied on MIR tasks [1]. Music can indeed be interpreted as a language, and automatic music generation has been a showcase for the NLP technologies in MIR. Among these techniques, Transformer models [2] have succeeded in complex tasks related to language understanding, overcoming the performances of more established architecture such as Recurrent Neural Networks (RNN) when huge amounts of data is available [3,4].

Jazz improvisations are good candidates for music generation, because their structure has limited need of long-term memory: unlike other kinds of compositions, a jazz musician is unlikely to repeat structures that s/he has improvised a long time before. However, complex phrasings, unusual rhythmic structures and sudden harmonic changes are challenging traits when training a music model.

In this paper, we introduce MINGUS [1] (Melodic Improvisation Neural Generator Using Seq2seq), a transformer architecture for modelling and generating monophonic jazz melodic lines. MINGUS handles pitch and duration as separate features, using two distinct transformer models. In addition, it exploits the whole available information by conditioning on other musical features, such as harmonic structure and rhythmic properties.

The remaining of this paper is structured as follows. After pointing some related work in Section 2, we will describe MINGUS in Section 3. Section 4 reports about an evaluation experiment, whose results are discussed in Section 5. In Section 6, we carried on a qualitative evaluation with a user survey. Finally, conclusions and future work are outlined in Section 7.

2. STATE OF THE ART

Data representation Musical data can be represented symbolically with different levels of abstraction and precision, involving features such as pitch and duration of the notes, relative position in the bar, harmonic structure, intensity (velocity), timbre. Different approaches have been experimented in literature for duration representation, among which time-step encoding [2] [5–8], note duration encoding [5,9,10] and note beat position encoding [5,11]. The duration information can be also modelled as a sequence and independently learned [9,12].

Model architecture The simplest approach for monophonic music generation is jointly learning the dependencies between features. This has been implemented in different architecture, such as RNNs [6], Generative Adversarial Networks (GAN) [5], combinations of GAN and RNN [13] or Transformer models [14].

An alternative strategy is to train separately to learn specific features of the data, then conditioning them on the other features. In [15] and [16], two LSTM models are trained separately on pitch and duration of the notes in the melodies. LSTM are also used in [9], in which different conditioning combinations – inter-conditioning between pitch and duration, chord, next chord and relative position in the bar – are compared. In **BebopNet** [12], a unique embedding representation of pitch and duration feeds a unique Transformer module. **SeqAttn** [6] obtained good performances using a modified conditioned LSTM attention unit.

Transformer-based architecture can be used for overcoming the problem of vanishing gradient of RNNs [17]. In polyphonic music generation, training transformer models on massive amounts of data produced impressive re-

[1] Named in honour of Charles Mingus (1922 – 1979), American jazz composer, double bassist and pianist.

[2] Sampling over time and using using a sustain character (s) for pitch continuation.

sults, as in OpenAI's MuseNet [3] and Magenta's Music transformer [4].

Evaluation Methods Evaluating the performance of a generative task is an open problem, with several metrics and methods proposed. Classical **machine-learning metrics** – **loss** and **accuracy** – are applied in evaluating music generation from a sequence-modelling point of view, measuring the capability of predicting the most probable class (e.g. pitch) given the sequence of all the previous ones. It could be argued that the purpose of generation tasks goes beyond the completely accurate prediction of the next token and so these metrics can capture only partially the quality of music generation systems. Nevertheless, they are useful to make a comparison among models.

Common metrics for generative NLP task evaluation can be used also for music generation, such as **perplexity** [18] and **BLEU** [19]. The latter in particular has been applied to music generation as a measure of similarity between two corpus of music [9].

Other metrics have been proposed for measuring how realistic a generated melody is by comparing it with the training corpus in musical terms. In [5], the authors propose a collection of metrics, which includes counting pitch repetitions, rhythmic variations and measuring harmonic consistency. Similar features are involved in **MGEval** [3], which computes the degree of similarity (KL-divergence) between two corpora of MIDI files by extracting the distribution of each metric on a reference corpus from the original data and on a corpus of generated musical sequences [20]. More metrics are proposed in [14], focusing on the structural coherence of the generated musical phrases. Other common metrics are purely music-related, such as **harmonic coherence** [5, 12], the measurement of the percentage of chord and scale tones among the generated notes. Finally, **focus groups and user surveys** have been extensively used for qualitative evaluation [7, 8].

3. APPROACH

In this section, we will describe in detail MINGUS, focusing on the strategy for data representation and its architecture.

3.1 Data representation

The required input formats are *MusicXML* or *abc* notation. We require that the chords – when available – are explicitly expressed by its signature in the right place in the measure [4].

Each melodic line is represented as sequences of the following features – with the range of possible values reported in square brackets:

1. **Pitch (P):** pitch of each note, as MIDI pitch number. Rests are represented with the character `R` [0-128]
2. **Duration (D):** duration of each note [0-12]
3. **Chord (C):** current chord in the starting beat of the note [0-128 x 4]
4. **Next Chord (NC):** next chord in the progression [0-128 x 4]
5. **Bass (B):** current bass in the beat the note starts on [0-128]
6. **Beat (BE):** number of beat in the measure the note starts on [0-3]
7. **Offset (O):** offset of the note from the start of the measure [0-95]

An example of input format for note sequences can be seen in Figure 1. The duration value D is extracted by sampling each measure into 96 equally sized parts and assigning to each note the closest duration from a dictionary of possible duration values chosen in advance; for example, the "quarter note" value is assigned to notes whose duration is closer to $d_{measure}/96 * 24$, where $d_{measure}$ is the duration of the measure in seconds. In this specific case the choice to divide a measure into 96 equal parts allows for representation precision up to 8th note triplets and dotted 16th notes. By using a greater number of samples it would be possible to represent more precisely many different duration values. This method of time division ensures flexibility to different music styles which could require the use of more complex time divisions such as quituplets of septuplets. Chords (C and NC) are always represented by their four fundamental notes in MIDI encoding, as already seen in [9, 12]; chords with more than four notes have been cropped, the VII degree has been added to chords with less then four notes.

Given that language models are normally trained on batches of phrases with a maximum fixed length, we included a melody segmentation strategy for dividing tracks into meaningful musical phrases. For this purpose, we consider a melodic phrase to end when a long rest – longer than a threshold r, equivalent to a quarter note triplet – is encountered or when the maximum sequence length $l = 35$ is reached. The threshold for the long rest duration and the sequence length have been chosen experimentally. It has been found that a different choice of maximum duration, if not extreme, do not have much influence on the result, while the sequence length has a greater effect on the result. Another observation is that the long rests at the end of each segment must be included into the sequence, otherwise the model will not learn to include them. Segments shorter than 35 tokens are padded with specific "pad" tokens.

3.2 Model Architecture

MINGUS is structured as two parallel transformer models with the same structure, respectively predicting pitch and duration, composed by the sole encoder module with a forward mask and a pad mask. This structure was chosen because it allows to capture the rhythmic variation with great

[3] The MGEval toolbox – which we use in this work – is available in its original implementation at `https://git.io/mgeval`

[4] Future work includes the possibility of automatically inferring the chords from the played notes.

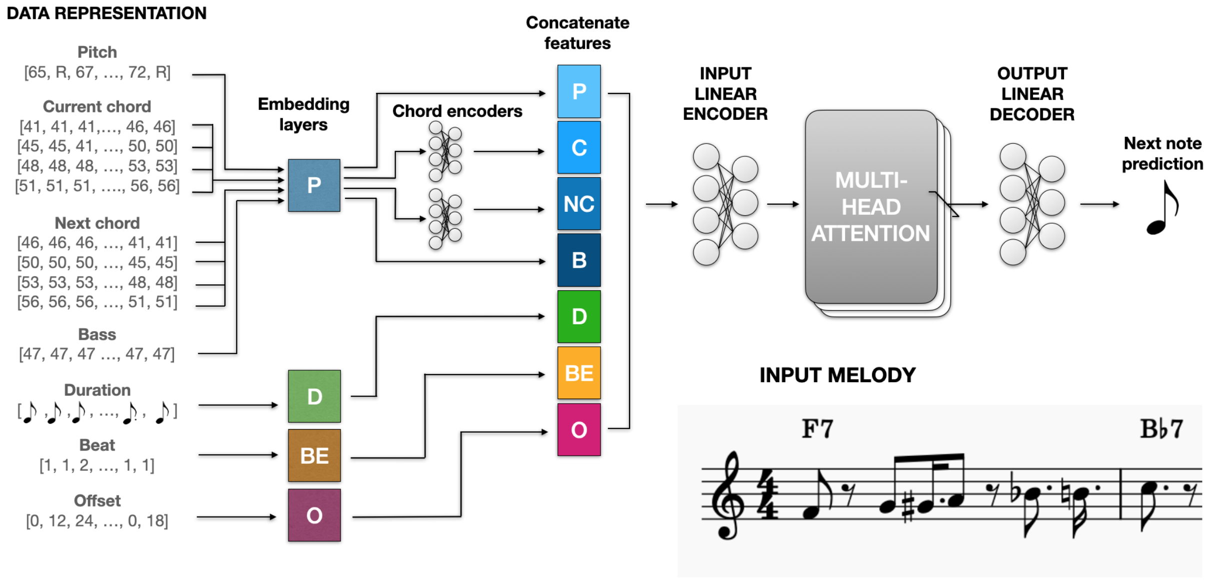


Figure 1: MINGUS model architecture and data representation. The example melodic sequence is extracted from Charlie Parker's improvisation on Billie's Bounce, Weimar Jazz DB

precision, by allowing the model to learn different embedding and weights for pitch and duration prediction. The architecture used in this experiment is shown in Figure 1. The structure is the same for pitch and duration models, the only difference is the output of the prediction, which can be either from the pitch or the duration dictionary.

Batched data is encoded with feature-specific embedding layers. Pitch-related data (melody pitch, chord pitches, next-chord pitches and bass) are encoded with a pitch embedding layer while duration, offset and beat have their own embedding dictionary. After embedding, chord pitches are grouped in a linear layer which is then combined with the other embedded features and fed into a four-layer, four-heads self-attention module.

The model was conditioned on all the features mentioned in Section 3.1. We performed an ablation study in order to understand the contribution of each feature, choosing the combination maximising the accuracy score. In particular, the optimal combination for pitch model included features D, C, B, BE, and O, while for the duration model B, BE, and O. However, it should be noticed that the feature combination that maximises accuracy might not be the one that generate the most convincing music samples. More results about this ablation study are included in the supplementary material.

4. EXPERIMENT

MINGUS was trained on two different datasets to evaluate its adaptability to different styles of music and compare it with other state of the art models. The **Weimar Jazz Database** (WjazzDB) [21] is a collection of annotated transcriptions of jazz solos, composed of 456 improvisations on famous jazz standards. It is a very diverse set of improvisations played on multiple instruments, including multiple jazz styles with different degrees of complexity. The dataset is complete with chords and bass information. The **Nottighman Database** (NottinghamDB) is a collection of 1034 folk songs [5] . The harmony of the music in this dataset is less complex with respect to the Weimar Jazz DB, nevertheless its smaller dimensions could be useful to show how the size of the dataset influences the generation results.

Each dataset was split into three subsets for training (70%), validation (10%) and testing (20%). The 35-token sequences were grouped into batches of 20 melodies for training and 10 melodies for validation and testing.

The network is trained for next-token prediction task using sequential information. Both pitch and duration are trained using cross-entropy loss function and Stochastic Gradient Descent optimizer. More details on the training parameters are available in the supplementary material.

Music generation is done by sampling the trained network given an input note sequence of variable length. The input melody is split into pitch and duration sequences and each sequence is given as input to the respective trained model. The output of the model are the probabilities for each token in the dictionary to be the next token. The most probable token is selected and added to the original sequence, which is then given back as input to the model, together with other required features for conditioning – features 3–7 in Section 3.1. This process is repeated for as many times as the number of notes to be generated, which of course must be the same for pitch and duration. After generating the new sequences of pitch and duration separately, they are combined and exported to MIDI.

[5] `https://github.com/jukedeck/nottingham-dataset`

5. RESULTS

This section reports the performance of MINGUS and compares it to SeqAttn [6] and BebopNet [12]. These two models have been chosen because they represent different state-of-the-art architecture for music generation task. BebopNet is based on transformer and SeqAttn is a bidirectional LSTM model conditioned on chords, with different features and duration representation. To obtain comparable results MINGUS [6] , BebopNet and SeqAttn have all been re-trained on the two datasets [7] , and evaluated on the same metrics.

The perplexity and accuracy of SeqAttn have been computed using the functions available in the authors' implementation. However, the prediction of the sustain token `(s)` is considered accurate even if the note that is being sustained is not correct; similarly, `(s)` is considered as a distinct token in the computation of the perplexity. Instead, in MINGUS and BebopNet duration is represented with a separate dictionary and this allow to have note-specific perplexity and accuracy.

5.1 Perplexity

The values of perplexity of the three models computed on the test set are collected in Table 1. For MINGUS, it is reported for pitch and duration models, while for BebopNet and SeqAttn it is computed on the summed entropy of pitch and duration.

Perplexity	MINGUS		BebopNet	SeqAttn
Dataset	**pitch**	**duration**		
WjazzDB	11.01	4.14	44.70	3.71
NottinghamDB	11.03	1.88	13.46	*1.40*

Table 1: Test perplexity scores for MINGUS, BebopNet and SeqAttn. Values in *italic* are reported from the original paper

The perplexity can give a general idea of the degree of the uncertainty of the model when predicting next token, it is however not a good indicator of music generation quality. The perplexity of all models changes according to the dataset. The greatest perplexities have been obtained for all models on the WjazzDB, despite the fact that Nottingham DB has fewer data, proving that the complexity of the music is an important factor for language modelling tasks.

5.2 Accuracy

The accuracy measures how many times the model prediction for the next note is correct. This metric could be useful to have an overall representation of the model performance but it does not guarantee a realistic music generation. The accuracy values of MINGUS and SeqAttn on all datasets are reported in Table 2. The implementation of BebopNet is not providing the computation of accuracy on a test set, therefore its scores are not shown in the table.

Accuracy [%]	MINGUS		SeqAttn
Dataset	**pitch**	**duration**	
WjazzDB	16.32	32.34	74.43
NottinghamDB	35.82	76.62	*90.26*

Table 2: Test accuracy comparison between MINGUS and SeqAttn. Values in *italics* are reported from the original paper

It is difficult to compare the accuracy results because of the different note representation and of the division of pitch and duration models in MINGUS. The model achieves a higher accuracy on the duration prediction and lower accuracy in pitch predictions; this could be due to the different size of vocabulary, but it could also be related to the difference between the two tasks, with an higher difficulty for pitch prediction. When comparing MINGUS and SeqAttn it should be considered that a percentage of the accuracy of SeqAttn is due to prediction of common sustain tokens.

5.3 MGEval

MGEval is a collection of metrics specifically proposed for evaluation of generative music tasks [20]. For MINGUS and BebopNet, we computed MGEval metrics comparing 15 reference tunes randomly selected from the original dataset – used as reference corpus – and a set of 15 tunes, generated from the same input by each model. SeqAttn generates music from an internally selected songs from the dataset seen during training, instead of accepting a track in input for triggering the generation; for this reason, MGEval metrics for SeqAttn are comparing the whole output of the model and the whole studied corpus (reference corpus).

Table 3 collects the results obtained on MINGUS, BebopNet and SeqAttn trained on WjazzDB. MGEval metrics yield very diverse results and do not reveal a clear overall winner. While the LSTM-based model (SeqAttn) has better scores on avg IOI and comparable results on pitch range and total used pitch, transformer-based ones (MINGUS and BebopNet) largely over-perform it in total pitch class histogram and note length histogram. MINGUS stands out in pitch class transition matrix and total pitch class histogram, whose result are largely better than the other two models. We interpret this result with a better modelling capability for transitions between notes, maybe due to MINGUS' flexible duration vocabulary. On the other hand, SeqAttn performs much better on NottinghamDB [8] . This suggests the duration representation employed in SeqAttn does a better job in generalising on the music style of NottinghamDB, while on WjazzDB the additional information provided in MINGUS and BebopNet improve generation quality.

[6] In the best configuration obtained from the ablation study

[7] WjazzDB has been converted from the original csv format into musical formats compatible with the studied implementations, namely into musicXML – for BebopNet and MINGUS – and into MIDI – for SeqAttn. 28 songs have been removed from the original dataset due to incompatibility with BebopNet, which was not recognising chords not in its internal dictionary; all models have been trained on this reduced version.

[8] Results provided in the supplementary material

MGEval	MINGUS		BebopNet		SeqAttn	
Measure	**KL div**	**overlap area**	**KL div**	**overlap area**	**KL div**	**overlap area**
total used pitch	0.172	0.7959	**0.007**	0.539	0.068	0.735
total used note	0.071	0.678	**0.046**	0.794	0.169	0.239
avg IOI	0.054	0.625	0.219	0.842	**0.049**	0.719
avg pitch shift	**0.041**	0.821	0.160	0.424	-	-
note length histogram	0.283	0.507	**0.054**	0.821	0.241	0.468
total pitch class histogram	**0.088**	0.864	0.137	0.786	0.405	0.658
note length transition matrix	**0.149**	0.695	0.210	0.850	0.261	0.388
pitch class transition matrix	**0.038**	0.836	0.118	0.737	0.183	0.744
pitch range	**0.037**	0.844	0.093	0.571	0.062	0.702

Table 3: MGEval comparison between MINGUS, BebopNet and SeqAttn on WjazzDB

5.4 Harmonic coherence

The harmonic coherence measures how many notes of each solo are coherent to the related harmonic context. It is defined here as the percentage of generated notes that are tones of the current chord, or of the scale associated to it. These metrics have been calculated on the generated tracks and on the entire original dataset. The results are reported in Table 4.

Harmonic coherence [%]	Chord	Scale
Original	49.17	72.16
MINGUS	**51.81**	**77.49**
BebopNet	40.66	64.55
SeqAttn	35.92	60.26

Table 4: Harmonic coherence on WjazzDB

These results confirm that MINGUS generated melodies tend to have greater harmonic coherence than other models, with BebopNet obtaining slightly worse performance and SeqAttn being less good on this metric. We may conclude that the conditional LSTM module proposed in SeqAttn is less able to capture the complex relationship between chords and melody with respect to Transformer-based architectures. Another reason may be identified in the presence of additional features such as duration and offset – in both MINGUS and BebopNet – which are beneficial for the harmonic coherence.

6. QUALITATIVE EVALUATION

6.1 Blind quiz

In order to evaluate our system from a user point of view, we performed a survey (blind quiz) involving listeners with different musical background and education. All the melodies have been exported into audio tracks and completed with a shuffle drum beat and chords for harmonic and rhythmic context.

Users were asked to rate a set of 15 short melodies (with average duration of 20 seconds) with a score from 1 to 5 based on how much they liked it. The set was composed 5 original melodies from the Weimar Jazz DB, 5 generated by MINGUS and 5 generated by BebopNet [9]. Users were unaware of which melodies were original and which ones where generated. The web app used for the quiz is available at [TO BE DISCLOSED AFTER BLIND REVIEW]. Figure 2 reports the obtained scores, detailed for 3 categories of users: music lover (8 participants), music student (9), professional musician (11).

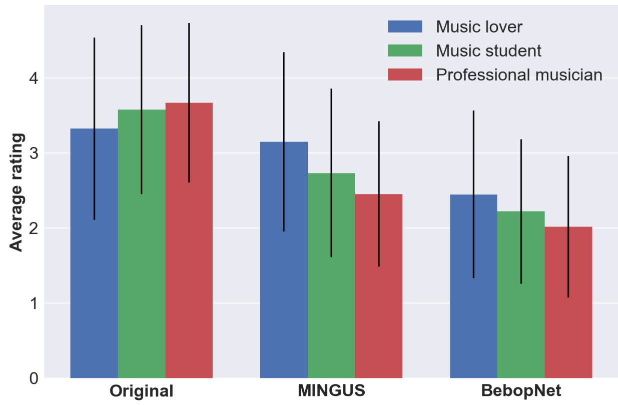


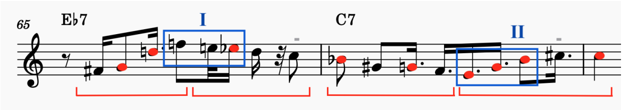
Figure 2: User evaluation summary

As expected, listeners are capable of identifying the original musical phrases with different degrees of confidence, proportional to their level of musical expertise. Overall the evaluation pointed out that MINGUS generations tend to be preferred by the users with respect to BebopNet generations, probably due to the greater harmonic coherence which makes the melodies more pleasing to the ear. There is still a clear difference between machine learning generated samples and original ones, especially when evaluated by high-skilled musicians. It should also be pointed out that this kind of evaluation takes into account very short, selected music segments: the difference between machine-generated and original samples may probably be more evident on long tracks.

[9] The chosen melodies are available in the supplementary material

6.2 Musical insight on the generations

Taking a look at the generated tracks in musical terms could be useful to identify areas of improvement. In this section we propose a musical analysis of a phrase generated by MINGUS in comparison with an original phrase. The two phrases are shown in Figure 3.



(a) Original improvised phrase

(b) MINGUS generated phrase

Figure 3: Comparison of original and generated musical phrases on Blues for Blanche by Art Pepper, Weimar Jazz DB. Chord tones in the melody are highlighted in red.

Pitch and duration The figure highlights in red the notes that are part of the underlying chords. In the original improvisation, the red notes are more frequent and tend to have a longer duration. The last note of this phrase is indeed a note of the chord, a typical choice by jazz improvisers because it creates a feeling of tension release. On the other hand, in the phrase generated by MINGUS few importance is given to chord tones. We can also observe that notes in phrase 3a present more homogeneous and repetitive duration patterns, while in phrase 3b note durations – although not far from each other – do not follow any specific pattern.

Patterns It is possible to spot some patterns in the phrases construction, highlighted in the figure by red lines. In the original one we can see two clear upward and downward motions as the phrase progresses. Although MINGUS seems to have grasped a general idea of such behaviour, the note movement in the generated phrase is not very clear. Other interesting segments are highlighted in blue. In the first blue segment of phrase 3a the melody is out of tune, but this is justified by chromatic downward motion in the melody. In the second blue segment the improviser performed a $C7$ arpeggio to end the phrase. An interesting similar behaviour appears the blue segment of phrase 3b, where also MINGUS performs an arpeggio at the end of the phrase. Unfortunately in this case it is a $C\#m7$ arpeggio, which is out of tune in the key of $Ab7$, so the result is not quite as pleasing.

Overall, MINGUS has learned to generate musical phrases separated by longer rests with approximate upward and downward motion and approximate harmonic coherence. Nevertheless, the generated phrases still lack a strong internal structure and the typical call-and-response phrase behaviour of jazz solo phrases, with few connections from one generated phrase to the other.

7. CONCLUSIONS

MINGUS uses a transformer architecture to generate music by separately predicting pitch and duration. The model was experimented on two different popular datasets and evaluated at different levels using a broad range of metrics, revealing comparable performances with respect to the state of the art. The experiment proved the capability of transformers to model and generate realistic melodic lines in the style of a jazz improvisation, with harmonically better results than LSTM. The MINGUS architecture proved to be particularly good at obtaining harmonically coherent melodies. An implementation of MINGUS is available in open-source at [TO BE DISCLOSED AFTER BLIND REVIEW].

The choice of metrics has a crucial impact on the evaluation of generation models, making it necessary to use many metrics at different levels of abstraction to obtain a reliable quality estimation.

During the experiments, the conditioning features have shown to learn different hidden representation of the data, which brings to different models. These learned models should not necessarily be ranked in a better-worse scale, but can be considered as alternative sounding. In future work, we intend to further measure the impact of the different features, in order to enable an aware use for generating specific styles, and to explore conditioning on other features provided by WjazzDB, such as instrument, jazz style and rhythm feel. In addition, we want to explore the role of *temperature control* as in [12, 22] , for tuning the system towards a more *creative* generation.

Even though MINGUS has been designed and trained specifically for music modeling and generation, we intend to improve and adapt it for other MIR tasks such as score music classification, bass line generation, automatic harmonisation, assisted composition, automatic music interpretation, conditional regression of musical features. Further research must be carried on for improving music generation systems in order to achieve long-term phrase-level coherence and to be applied in live condition, including interacting with musicians for educational and artistic purposes.

8. REFERENCES

[1] S. Oramas, L. Espinosa-Anke, S. Zhang, H. Saggion, and X. Serra, "Natural Language Processing for Music Information Retrieval (Tutorial)," in *17^{th} International Society for Music Information Retrieval conference (ISMIR 2016)*, New York, USA, 2016.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017. [Online].

Available: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[3] C. Payne, “Musenet,” 2019. [Online]. Available: https://openai.com/blog/musenet/

[4] C.-Z. Anna Huang, A. Vaswani, J. Uszkoreit, I. Simon, C. Hawthorne, N. Shazeer, A. M. Dai, M. D. Hoffman, M. Dinculescu, and D. Eck, “Music Transformer,” in *International Conference on Learning Representations (ICLR)*, New Orleans, USA, 2019. [Online]. Available: https://openreview.net/forum?id=rJe4ShAcF7

[5] N. Trieu and R. Keller, “JazzGAN: Improvising with Generative Adversarial Networks,” in *6^{th} International Workshop on Musical Metacreation (MUME)*, Salamanca, Spain, Jun. 2018, p. 8. [Online]. Available: https://doi.org/10.5281/zenodo.4285166

[6] J. Jiang, G. Xia, and T. Berg-Kirkpatrick, “Discovering Music Relations with Sequential Attention,” in *1^{st} Workshop on NLP for Music and Audio (NLP4MusA)*. Online: Association for Computational Linguistics, 16 Oct. 2020, pp. 1–5. [Online]. Available: https://www.aclweb.org/anthology/2020.nlp4musa-1.1

[7] F. T. Liang, M. Gotham, M. Johnson, and J. Shotton, “Automatic Stylistic Composition of Bach Chorales with Deep LSTM,” in *18^{th} International Society for Music Information Retrieval Conference (ISMIR)*, S. J. Cunningham, Z. Duan, X. Hu, and D. Turnbull, Eds., Suzhou, China, 2017, pp. 449–456.

[8] O. Peracha, “Improving Polyphonic Music Models with Feature-Rich Encoding,” in *21^{st} International Society for Music Information Retrieval Conference (ISMIR)*, Online, 2020. [Online]. Available: https://program.ismir2020.net/poster_2-01.html

[9] B. Genchel, A. Pati, and A. Lerch, “Explicitly Conditioned Melody Generation: A Case Study with Interdependent RNNs,” in *7^{th} International Workshop on Musical Meta-creation (MUME)*, Charlotte, NC, USA, 2019.

[10] F. Carnovalini and A. Rodà, “A Multilayered Approach to Automatic Music Generation and Expressive Performance,” in *2019 International Workshop on Multilayer Music Representation and Processing (MMRP)*, Milan, Italy, 2019, pp. 41–48.

[11] E. P. Nichols, S. Kalonaris, G. Micchi, and A. Aljanaki, “Modeling Baroque Two-Part Counterpoint with Neural Machine Translation,” in *International Computer Music Conference (ICMC 2020)*, Santiago, Chile, 2020. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02940164

[12] S. H. Hakimi, N. Bhonker, and R. El-Yaniv, “BebopNet: Deep Neural Models for Personalized Jazz Improvisations,” in *21^{st} International Society for Music Information Retrieval Conference (ISMIR)*, Online, 2020.

[13] O. Mogren, “C-RNN-GAN: A continuous recurrent neural network with adversarial training,” in *Constructive Machine Learning Workshop (CML) at NIPS 2016*, Barcelona, Spain, 2016, p. 1.

[14] S.-L. Wu and Y.-H. Yang, “The Jazz Transformer on the Front Line: Exploring the Shortcomings of AI-composed Music through Quantitative Measures,” Online, 2020.

[15] J. Franklin, “Jazz Melody Generation from Recurrent Network Learning of Several Human Melodies,” *International Journal on Artificial Intelligence Tools*, vol. 15, no. 04, pp. 623–650, Aug. 2006. [Online]. Available: https://doi.org/10.1142/s0218213006002849

[16] F. Colombo, S. Muscinelli, A. Seeholzer, J. Brea, and W. Gerstner, “Algorithmic Composition of Melodies with Deep Recurrent Neural Networks,” in *1^{st} Conference on Computer Simulation of Musical Creativity*, Huddersfield, UK, 06 2016.

[17] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training recurrent neural networks,” pp. 1310–1318, June 2013.

[18] N. Ranjan, K. Mundada, K. Phaltane, and S. Ahmad, “A Survey on Techniques in NLP,” *International Journal of Computer Applications (IJCAI*, vol. 134, no. 8, pp. 6–9, January 2016. [Online]. Available: http://doi.org/10.5120/ijca2016907355

[19] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “Bleu: a method for automatic evaluation of machine translation,” in *40^{th} Annual Meeting of the Association for Computational Linguistics (ACL)*. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, Jul. 2002, pp. 311–318. [Online]. Available: https://www.aclweb.org/anthology/P02-1040

[20] L. Theis, A. van den Oord, and M. Bethge, “A note on the evaluation of generative models,” in *International Conference on Learning Representations (ICLR)*, Apr 2016. [Online]. Available: http://arxiv.org/abs/1511.01844

[21] M. Pfleiderer, K. Frieler, J. Abeßer, W.-G. Zaddach, and B. Burkhart, Eds., *Inside the Jazzomat - New Perspectives for Jazz Research.* Schott Campus, 2017.

[22] S. Oore, I. Simon, S. Dieleman, and D. Eck, “Learning to Create Piano Performances,” in *NIPS 2017 Workshop on Machine Learning and Creativity*, Long Beach, USA, 12 2017.