Generative Adversarial Neural Networks for Random and Complex Chord Progression Generation

Alexander Melendez-Rios
Universidad Peruana de Ciencias Aplicadas (UPC)
Lima, Peru
u201815920@upc.edu.pe

Roberto Vega-Berrocal
Universidad Peruana de Ciencias Aplicadas (UPC)
Lima, Peru
u201210455@upc.edu.pe

Willy Ugarte
Universidad Peruana de Ciencias Aplicadas (UPC)
Lima, Peru
willy.ugarte@upc.pe

***Abstract*—In the latter years, recurrent neural networks for generative modelling of sequences with long range dependencies have been outperformed by the use of autoregressive models such as Transformer XL. With these models, a complete processing of long range dependencies has been increasingly improved and optimized in terms of memory and execution time. Our experiments use Google BERT, a transformer-based machine learning technique for natural language processing that can produce high fidelity text generation outputs. We use this technique to work on music, as this is represented by symbols and can be considered a type of language. Jazz oriented or influenced music composition often includes dealing with harmonic progressions in which each chord usually consists of four or more different notes. To design complex chord progressions is a very specialized task and expensive in time that demands lots of hours of music theory studying. Furthermore, currently there aren't apps and music industry plugins designed to generate this type of output data interactively. To work on this we trained a generative adversarial neural network that operates using two Transformer XL as classifier and generator respectively. We trained the model on a self designed corpus of MIDI files that explicitly considers the harmonic patterns of various well known Jazz themes, leaving behind lead melody and rhythm patterns. After achieving an stable training of the model we were able to generate extensive batches of complex harmonic progressions on MIDI files. These outputs clearly expose stylistic properties that are present in Jazz music chord progressions.**

***Index Terms*—TransformerGAN, Jazz, Generative Music, Symbolic Music, Harmonic Progressions, Chord Sequence, Generative Adversarial Networks, Generative Modelling.**

I. Introduction

Music sequences are similar to natural language representations because they contain long range dependencies between their elements. These sets of dependencies can be contemplated in almost any musical sequence and they support the existence of two or more independent voices, also known as the melodic lines that occur simultaneously during a sequence.

Musicology research has observed and demonstrated that the execution of polyphonic music in oral traditions of indigenous groups is based on different patterns and implicit sets of rules that are a reference for creating or executing any type of music [1].

It has been observed in traditional vocal executions of Central Africa tribes that independent voices in their polyphony are executed in synchrony and exhibit two major techniques present in the study of academic music: counterpoint, where every voice is independent in terms of rhythm and melody and is always interacting with other voices, and homorhythm, where each voice occurs in parallel intervals [2].

In more general terms, polyphonic music is every simultaneous combination of two or more notes or melodic lines. [1]. A subset of it is homorhythmic music, where a set of melodic lines occur simultaneously with equal duration but in different pitch. That being said, harmonic progressions or chord sequences are homorhythmic music expressions that serve as the guiding base which sets the harmonic rules to be followed and used in the developing or execution of a song or composition.

Polyphony concept does not occur in the same manner for natural language representations because expressing two or more words simultaneously lacks of logic in a common conversation or message, unless it occurs as a literary device for narrating an specific situation where two or more people are talking.

Contrary to the principles of languages, in music, mutual inclusion does not necessarily exclude tones articulated at the same time, within the concept of articulated tone [3]. The radical opposite occurs in musical styles like Jazz, impressionism or any style that seeks a complex harmonic sound where at least four or five different notes can happen at the same time as a recurring event.

The task of generating sequences that show the above properties of long range dependencies has been already a matter of research and in recent years vast improvements have been done as a result of greater advances in data science and deep generative modeling algorithms. Before the apparition

[1] https://www.britannica.com/art/polyphony-music

of Transformer models, symbolic music generation was better performed by recurrent neural networks.

The performance of these networks exhibit clear deficiencies due to the inefficient memory handling of long-range dependencies. As a result, many details and patterns present in the training data distribution are lost. Recurrent neural network models can memorize dependencies that are nearby to specific elements at a certain step of the sequence, not dependencies that exist further than that.

The release of 'Attention Is All You Need' publication [4] introduces Transformer models, which outperform previous recurrent models in the task of managing long range dependencies, with unprecedented success.

A year later, Google research team released Music Transformer [5], pushing further the state of the art on symbolic music generation. A much more efficient and optimal management of long range dependencies has great impact on learning. After training Google's model on an extensive dataset of classical piano executions, the machine generated impressive symbolic music that mimics the virtuous playing of professional pianists, capturing the long dependencies present in polyphonic music.

Another great step that has had great impact on the task is a technique for symbolic music learning that was designed using BERT model, adapting it's natural language processing capabilities for symbolic music pattern extraction over MIDI files [6].

In a subsequent experiment, MusicBERT technique is used as the classifier component in an adversary neural network that is pre trained to then operate together with a Transformer network generator component. This experiment has achieved the best results to date. For this reason, we used TransformerGAN [7] as the starting point for conducting our own generative experiments on symbolic music, specifically complex chord sequence generation.

- We developed the ChordGAN model by retraining TransformerGAN with MusicBERT for conducting our own generative experiments on symbolic music, specifically complex chord sequence generation.
- We proposed to use ChordGAN as a starting point to work onto more complex compositions. As a recommendation for professional musicians and composers to increase their possibilities and paths when composing original music.
- We developed an API to facilitate the use of the model.

In Section II, we explore and discuss recent and similar approaches for symbolic music generation and their results. In Section III, we explain the main notions required to develop our work, such as the musical theory and generative modeling concepts. In Section IV, we detail all the experiments carried out and their results to prove the feasibility of our proposal. Finally in Section V, we describe the main conclusions and perspectives.

II. Related Works

Automated symbolic music generation has been a topic of research in recent years and has been positively impacted by a battery of deep learning advances that will be discussed in this section, along with different methods that were previously considered as the state of the art.

Before the emergence of transformer models, the state of the art for dealing with sequences with long range dependencies was the application of recurrent neural networks. One of the most well known and disruptive works for symbolic music generation through recurrent neural networks is DeepBach [8]. This implementation was trained on Johann Sebastian Bach chorale works and is capable of continuing and creating new chorales in a conditional manner: the user defines an input melody as unary constraint and the machine invents a piece of music applying the learnt set of contrapuntal and harmonic rules. These exist implicitly through the training data distribution.

Another application of recurrent neural networks is [9], which consists of a bidirectional model of Long Short Term Memory neural network that can generate rhythmic-melodic sequences of two minutes approximately. The system can learn the melodic and rhythmic patterns and their dependencies in order to generate novel music. Even so, it is appreciated that the sequences do not have a clear sense of harmony, causing them to be perceived as aimless or illogical in a diatonic musical sense.

The research that enabled the development of transformer models [4] exposes the first glimpse of the self-attention mechanism, which changes recurring layers with multi-headed self-attention. This optimizes the way recurrent neural networks deal with long range dependencies, where there is an inability of retaining information of every element in the sequence, for it manages to work only with the last hidden states of the decoder, losing import information of further elements and not only the most recent ones. The main difference with this, is that self attention mechanism focuses to pay attention not only on the last states of the encoder but on every single state. This characteristic enables to access information about all of the elements of the sequence being examined. Simply put, the self attention method that enables the transformer architecture eschews recurrence and relies on the mentioned self attention mechanism to examine global dependencies on the sequence being processed.

In [10] a successful experiment of applying the transformer architecture to generate symbolic music is carried out. This work achieves consistency across generations of long-running music with a parameterizable length up to three minutes. This work is able to generate symbolic music of thousands of steps with a logical structure, generating continuations that coherently elaborate over a given motif or pattern. With this solution arises a new problem: generation takes a long time and creating sequences longer than 3 minutes demands too much time as the complexity of the operation is quadratic.

In [7] an improvement for Music Transformer is proposed.

The training process occurs through the pre-training of a classifier, for it's frozen layers that were trained using SpanBERT model will serve as the discriminator starting point for a generative adversarial neural network training. This work used the MIDI Maestro dataset, which consist of virtuous classical piano executions. The proposed metric for generated data evaluation is negative log likelihood (NLL) and illustrates similarity between generated and training data. The closer to zero NLL value, the more similar the generated artificial data is to the training data distribution. In our experiments, the key differences residing in our custom training data are shown.

III. GANs and Chord Progression Generation

The stock management process carried out by storekeepers in supermarkets is practically manual and, therefore, quite laborious. The achievement of this task is linked to human capacity, which is why it is often not carried out satisfactorily. In addition, there are few technological tools that storekeepers present to facilitate the fulfillment of these tasks. This is due to the fact that there is a degree of difficulty in developing said technological tools based on object detection models, which not only allow the object to be identified, but also to account for and verify the status of various products.

A. Preliminary Concepts

The generation of harmonic progressions through artificial intelligence is a challenge within the field, since music has different variants in harmony and melody generation, genres, times, etc. Techniques based on GANs have allowed to solve, although not completely, problems of finding variants in music. Specifically for this work, which seeks to generate series of combinations of 4+ notes in a sequential manner, TransformerGAN technique resulted of great value to work within the scope of our proposal. It is important to clarify some key concepts to understand the music wise logic behind the experiment.

1) Music Notions:

a) Chord: : In simple terms, a chord is a combination of three or more notes or elements which is heard simultaneously. Depending on the harmonic style or intention, a chord may be consonant or dissonant. Consonance denotes calm and repose, while dissonance denotes movement and instability. The study of harmony observes in detail the implicit rules and patterns that arise from the combinations of notes in a sequential manner [11]. A commonly known symbolic representation of a chord can be seen in Fig. 1, where a simple C major chord is shown in vertical disposition of the notes, which demands executing all of them at the same time.

b) Harmonic Progression: : In the study carried out, the harmonic progressions deal with a succession of chords to order and keep the music coherent and controlled. Harmony represents an important dimension in music, since it allows a deep analysis of a composition in which traditional academic approaches give specific harmonic functions to different type of chords. An harmonic progression consists of many chords and their relationships in a sequential manner, creating and liberating tension through time [12]. A greater combinatorial problem arises when the chords contain more than four notes. This type of chords are known as extended chords and their execution increase the sense of complexity in music. This is a characteristic that occurs particularly in jazz music and in late romanticism music [3].

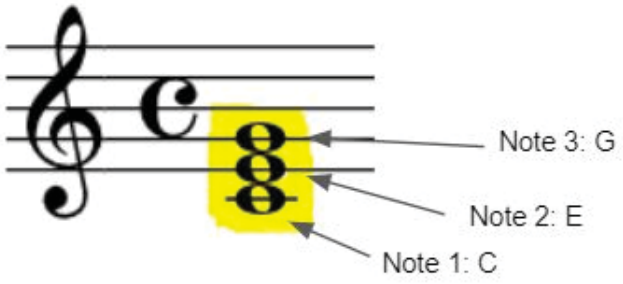


Fig. 1. Sheet music representation of a C major chord - Own elaborated

An example of a jazz influenced chord progression can be heard here[2]. As we can see in Fig. 2 every chord in the sequence relates implicitly to every other chord. These patterns are hidden and illustrate long range dependencies between every chord in the sequence.

c) Generative music: : The concept of generative music was created by Brian Eno, which refers to patterns or rules used to generate automated music. According to him, by parameterizing musical data, it can be used to generate random compositions, just as biological systems generate random events according to rules. Generative music may be the ability to generate new music output in real time, but it is not exclusively a computer based system. In [13], maintains Brian Eno, generative ideas were previously cumbersome, requiring proprietary hardware and software, until the smartphone was born, portable computer and hi-fi devices. Nonetheless, for now, generative music remains a fertile ground for future research and development, as it has yet to gain the everyday popularity of concerts and streaming durational music.

2) Generative Models:

a) Generative Adversarial Network (GAN): : Within the field of machine learning, generative adversarial networks are about networks that compete with each other for data generation with similar data from the training stages. A discriminating network and a generator are presented, the first is responsible for evaluating the results that the second generates, until it approves the accuracy of the output. All of this stems from the need to make machines capable of creating things by themselves, so that creativity works like the human brain and builds new things. The latter was an artificial intelligence problem that GAN has been able to solve. The GAN technique has reached branches of medicine, in [14] it is explained that it

[2]https://www.noteflight.com/scores/view/932b4a813a7bac45bf17e8e55438cf8d21dd43d0

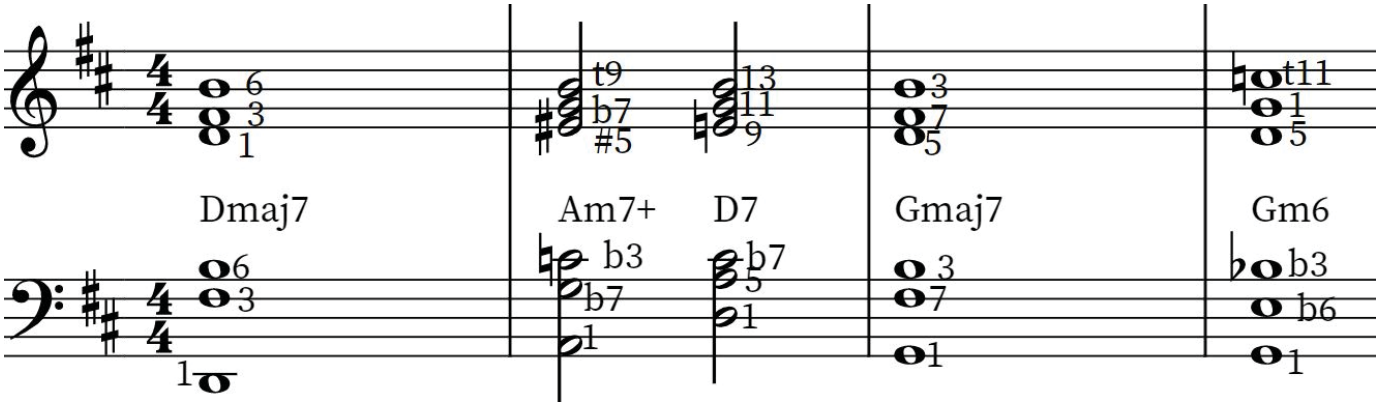


Fig. 2. Sheet music representation of jazz chord sequence involving five and six note executions per chord.

is capable of obtaining a new molecular representation, since the training logic could generate new compounds. This idea helps efficiency, time and costs during the drug design process in large companies.

b) Transformers XL: : At both the character and word levels, it is the first self-attention model to achieve substantially better results than RNNs. Maintains temporal consistency without breaking dependency learning beyond a fixed duration [15]. Recursion at the segment level and positional coding make up this system. By using this approach, long-term dependency is not only captured, but context fragmentation is also eliminated. Although it was intended for the creation of automated images, it is adaptable for music generation. In Fig. 3. Transformers are appreciated for their classical architecture, where encoders and decoders are differentiated, and convolutions and recurrences are not required to obtain outputs. An encoder section is displayed on the left of the graph, which represents the input stream in a continuous form, which then passes through a decoder section on the right. As a result of a masking stage, sequences of outputs are made (which makes it unidirectional), and it is possible to predict things based on these probability outputs.

c) BERT: : The term means Bidirectional Encoder Representation from Transformers. As opposed to recent models of language representation, BERT is used to condition left and right context across all layers in order to pretrain deep bidirectional representations from untagged text. With just one additional output layer, the BERT model can be tuned for a variety of tasks, including question answering and language inference, without modifying its architecture substantially task [16]. For BERT procedure there is a pre-training and fine-tuning process. In Fig. 4. there is an example where we can see both pre-training and fine-tuning. Initiating models for different downstream tasks uses the same pre trained model parameters. It is important to fine-tune all parameters. Input examples are prefixed with a special symbol [CLS], and questions and answers are separated by a special token [SEP].

d) Negative Likelihood Estimation: : An ensemble of random variables is called a pseudo-likelihood in statistics. The outcome is either a simplified estimate or an explicit

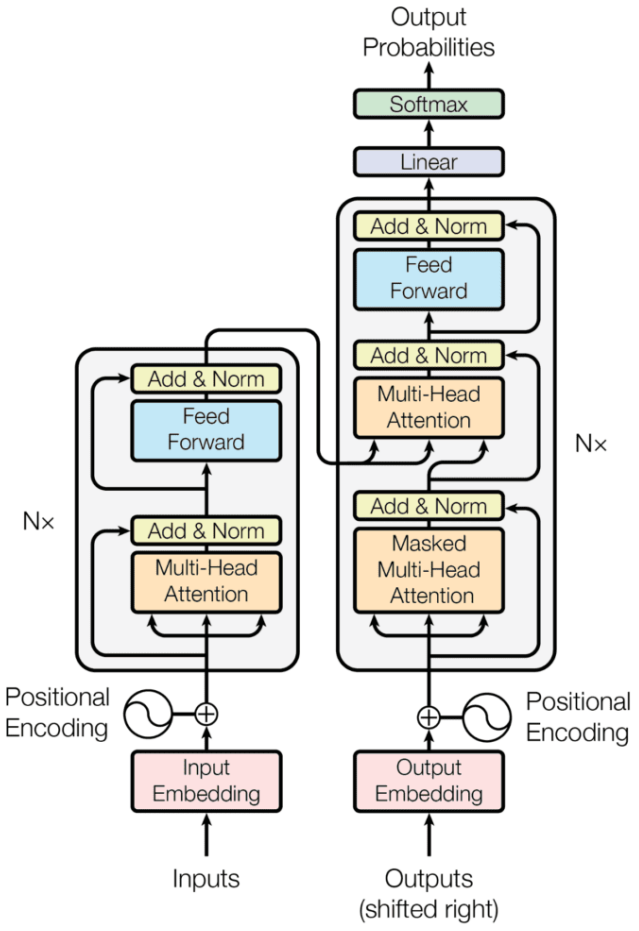


Fig. 3. Architecture of a transformer [4]

estimate of the inputs to the model, based on the computation of the probability function. As a function of the statistical mode parameters, the probability function represents a joint probability about the observed data. Continuous probability distributions and discrete probability distributions behave differently. When it comes to negative log likelihood, the density

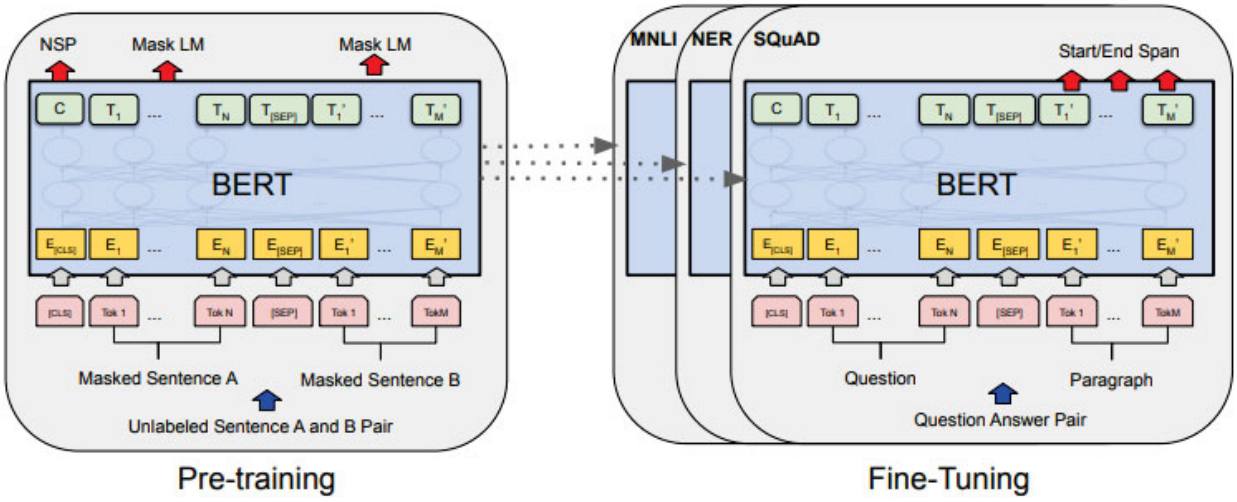


Fig. 4. Pre-training and fine-tuning of BERT [16]

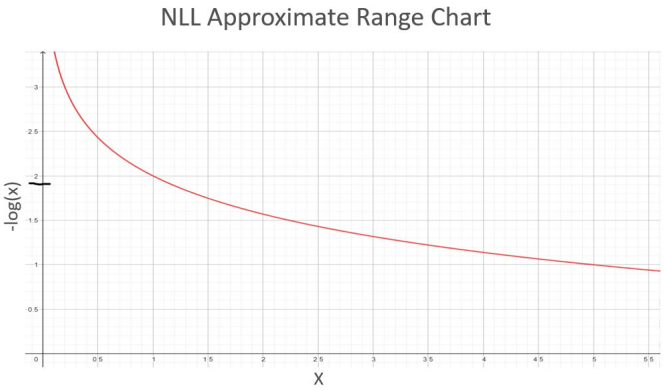


Fig. 5. Example of Negative Log Likelihood.

is said to be less than one, and it can be used as a loss function in Machine Learning. Its applications are present in many jobs and it serves as a metric to validate states of artificial intelligence training, such as noise removal in image generation [17]. For work purposes, the NLL indicator must approach the range of 0.5 and 1.9, as seen in the Fig. 5 there is a curve, and the closer the value is to 0, the more fidelity there is between the data generated with the training data. What is shown in the graph is plating as a simple example with the simple equation $f(x) = -\log_5(x) + k$

B. Method

The task of creating original music is a subjective process that involves technical knowledge of music theory in conjunction with the application of human creativity. The goal of this work is to automate the generation of chord sequences. A requirement is that every chord has to have a minimum of 4 notes. A user interface enables a TransformerGAN architecture that allows the end user to export generated MIDI outputs with no effort more than a simple input and a short waiting time. The resulting MIDI files can be easily imported in most of music production software commonly used for production and composition in the music industry. We believe that these experiment outputs could serve as creative input for human creators, which would evidence a path in arts where computational creativity assists human composition methods. We will validate this later in section IV.

We have designed and prepared a custom dataset named JazzHarm. This effort was made to achieve the main goal of our work: being able to randomly generate novel and complex chord progressions influenced by jazz harmony patterns on demand. These patterns exist implicitly in symbolic representation through music itself. For this purpose we chose a collection of jazz themes from [18], one of the main formal sources of knowledge from where musicians learn jazz music theory and execution of their melodic or/and harmonic instrument. These themes were recorded manually using a MIDI keyboard. The process of recording was focused to only capture the chord sequence of the song, not the music itself as a whole, which for the jazz style would commonly contain a lead melody, a rhythm pattern, a walking bass line, among many other possible independent voices that would complicate excessively the data patterns. An illustrative example of the first chord sequence recorded for JazzHarm can be appreciated in Fig. 6.

The chosen themes depict the execution of chords of 4+ notes, containing the previously mentioned higher-number-of-notes combinatorics scenario. A total of 25 themes were recorded. These include some bossa nova, big band themes and jazz ballads. The jazz chord symbols of a jazz ballad song was interpreted for our first entry on the dataset and can be seen in Fig. 7. Each one of these were read and recorded multiple times with the purpose of obtaining different possible variations of the chords. For example, on each re recording of a song different chord inversions were executed, more notes

were added to get an extended chord, or a passage was re harmonized. By constructing the dataset in this manner we made sure to register the implicit complexities of the selected composition harmonies, also to be perceived by the computer as long range dependencies between each of the notes of the sequences.

It is very important to remark that our dataset construction approach is music reductionist in the sense of handling only one dimension of music at a time. In this particular case, the only dimension considered is harmony. Nowhere in the data a non homorhythmic pattern is to be found, for we want the computer to generate pure chord progressions, not independent voices or melodic lines happening with different rhythmic patterns. We explicitly leave behind every music symbolism apart from harmony.

After recording the files, a n a ugmentation f unction [3] was used on our recorded data to increase ingestion volume on the model. This function permuted over the recorded symbolic MIDI representations through the twelve different tonal keys (C, C♯, D, D♯, E, F, F♯, G, G♯, A, A♯, B), extending to higher and lower octave notes, but with constraint limits to not use too low or too high notes. This processing resulted in 2430 symbolic music sequences out of the 25 previously selected compositions.

At this point, we were ready to begin the model pre training using MusicBERT technique. In contrast to the original TransformerGAN for symbolic music experiment, the whole training process was executed on a single Tesla P100 GPU. We only modified the input batch size of the network from 512 to 128, as the original experiment trained the model using four GPUs simultaneously. We pre trained the classifier on our custom data. We paused training after loss function threw a value around 0.547. At this point the classifier has learnt fairly enough to begin generator network training. With the classifier layers frozen and pre trained we then start TransformerGAN training. At this step we wait patiently to get a NLL value around 0.5 and 1.8. This would indicate that our model is able to generate artificial data similar to that contained within the input distribution. A value around 4.0 indicates that the output data is extremely different, while a value near 0.0 means the exact opposite. The first mentioned range of values worked great for our objectives, and further details will be illustrated in section IV.

After training successfully the generative adversarial neural network we implemented a simple API and interface that was thought for an intuitively interaction for modifying minimum model inference configuration parameters. This way the end user is able to generate chord sequences on demand through the presented stochastic process. In our final solution, the output sequences are exportable with the purpose of letting the end user to use them within a music production software.

[3]https://github.com/amazon-science/transformer-gan/blob/main/data/music_encoder.py

IV. Experiments

The process to achieve an stable chord progression generative model will be explained from work environment selection to training and querying of the model. We provide access to a repository of results and training dataset. In the same way, discussions of the work will be raised.

A. Experimental Protocol

Model training and generation will be detailed in order to illustrate the experimental process.

1) Development Environment: A total of 32GB of RAM and a Tesla T4, Tesla P100 GPU and Tesla K80 were available to us through usage of Google Colaboratory Pro plan. This were the tools that enabled us to train our Transformer model. Up to 200+ GB of space in Google Drive was needed to store network weights.

2) Dataset: The first experiments were made over a well known dataset named MIDI and Audio Edited for Synchronous Tracks and Organization (MAESTRO), which has about 200 hours of MIDI recorded over recordings through recent years of a well known annual piano competition. We couldn't obtain significant results due to training complications resulting in very discouraging results. Furthermore, the used dataset contains virtuoso performances, which isn't aligned with work's scope: generating plain chord sequences. We only used this dataset with the purpose to configure and obtain a first glimpse of artificial data, for it was used in original TransformerGAN for Symbolic Music experiment [7].

More directed training was made when a fair amount of chord sequence data was recorded. Our proposed dataset, JazzHarm, contains 92 interpretations of 25 jazz standard plain chord progressions. The music wise and technical chord notation for implementing this dataset was extracted from The Real Book. The selected songs consist mainly of jazz ballads and bossa nova themes. After recording these interpretations an augmentation function was used to permute through our data and obtain 2430 MIDI files. JazzHarm 1.0 version is available[4].

3) Models Training: Different experiments were executed using Google Colaboratory pro subscription over the course of five months. Colab virtual environment provided enough GPU RAM (32 GB) for Bert pre training and adversarial training. Dataset train, test and validation subsets were divided following a 8:1:1 proportion. Operating with Google Colaboratory took an estimation of 2 weeks and 10 hour for classifier pre training using JazzHarm data. During this process we saved classifier states but we only picked and kept four classifier network weights as we observed loss function was stabilizing upon a certain value.

When pre training threw a loss function of 0.348 we started executing main adversarial training on a parallel instance of Google Colaboratory while maintaining pre training instance running. Adversarial network training occurs faster compared

[4]https://drive.google.com/drive/folders/1Gca6tGh_N-SmN8Sxxk6ylOC-QBL5E4fR?usp=share_link

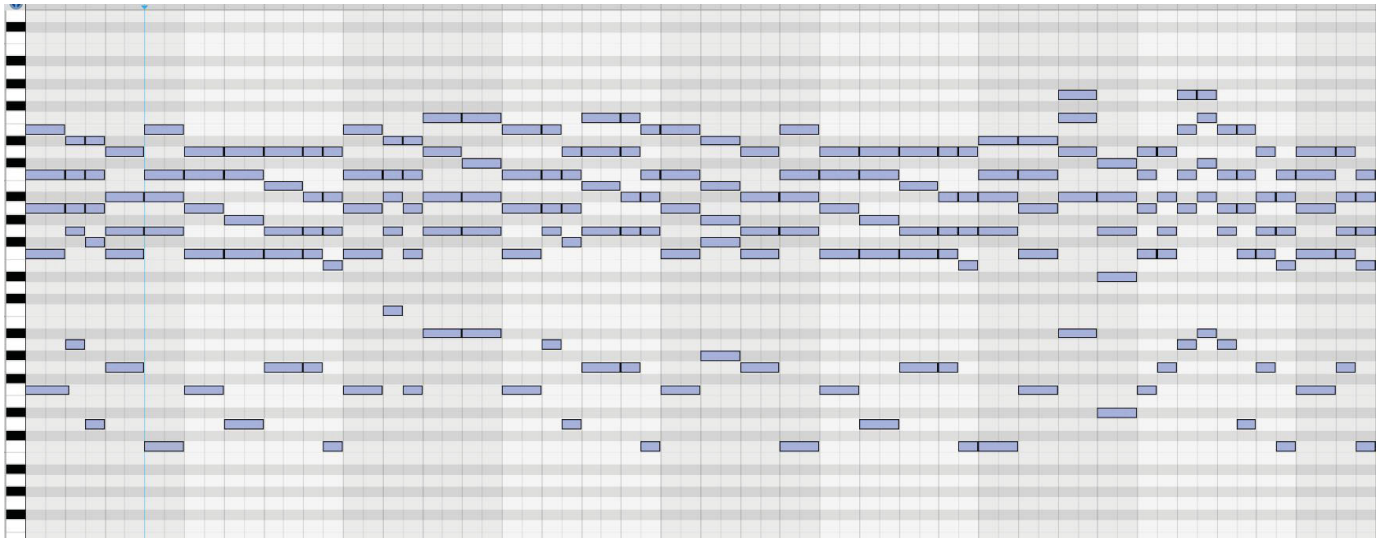


Fig. 6. The first sequence of our dataset is an execution of the harmony of 'A Foggy Day', a popular song by George Gershwin

Fmaj7 → Amb5 → D7b9 → Gm7 → C7
F6 → Dmb5 → G7 → Gm7 → C7
Fmaj7 → C-7 → F7 → Bb6 → Bbm6
Fmaj7 → Am7 → D7 → G7 → Gm7 → C7

Fmaj7 → Abm7 → Gm7 → C7
F6 → Dmb5 → G7 → Gm7 → C7
C-7 → F7 → Bb6 → Eb7
F6 → Gm7 → Am7 → Bbm6 → Am7 → Dm7 → Gm7 → C7

Fig. 7. The previous illustration on the first sequence included in our custom dataset is an interpretation of the shown chord sequence

TABLE I. RESULTS REPOSITORY ORGANIZED BY BERT CLASSIFIER PRE TRAINING LOSS FUNCTION VALUE

BERT Pre Training Loss	Number of Files Generated
.348	23
.452	470
.527	40
.547	550

to classifier pre training. Given the mentioned configuration and environment, execution time for each adversarial training from scratch over our data could be estimated to take between 8 and 10 hours. We ran several adversarial training for each of the four pre trained classifier weights. Every experiment evidence chaotic NLL evolution and when it stabilizes towards a good NLL value, the generated MIDI data depicts complex and jazz influenced chord progressions, as observed in the training data distribution.

4) Source Code: We used and adapted Amazon Science TransformerGAN code repository [5] to execute our experiments. This is the source code presented in TransformerGAN for symbolic generation original publication [7].

B. Results

After pre training the classifier we started adversarial training. During this process, Transformer GAN generator weights were saved for later generation testing.

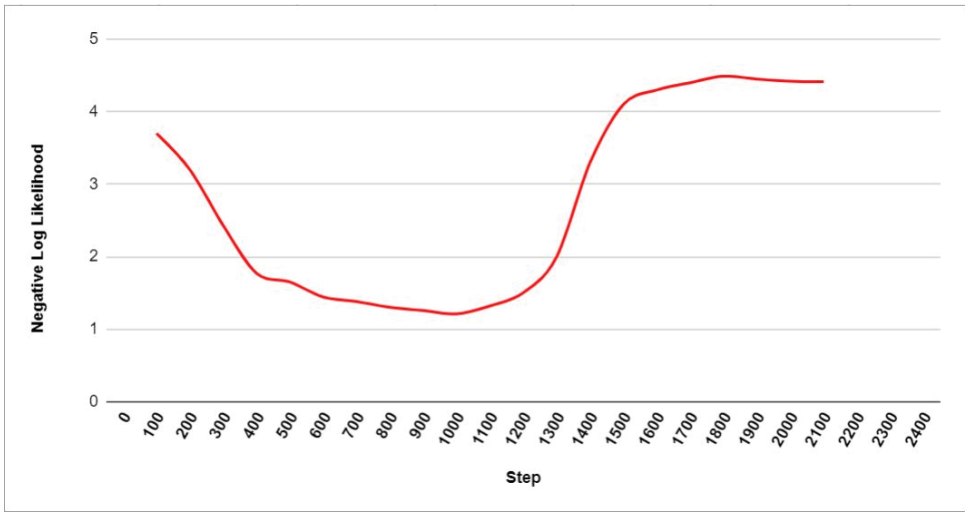
We queried different network weights for testing model inference capabilities. A wide collection of Transformer GAN outputs was curated using the resulting artificial data. This repository has been shared and is accesible [6]. In Tab. I shows information about the presented experimental repository. After hearing the generated data, our findings suggest that a valid value for a network state's negative log likelihood estimation would fluctuate between 0.5 and 1.8. It's important to have in mind that a value closer to zero depicts more similar patterns to training distribution data while a higher value, the opposite.

C. Objective Evaluation

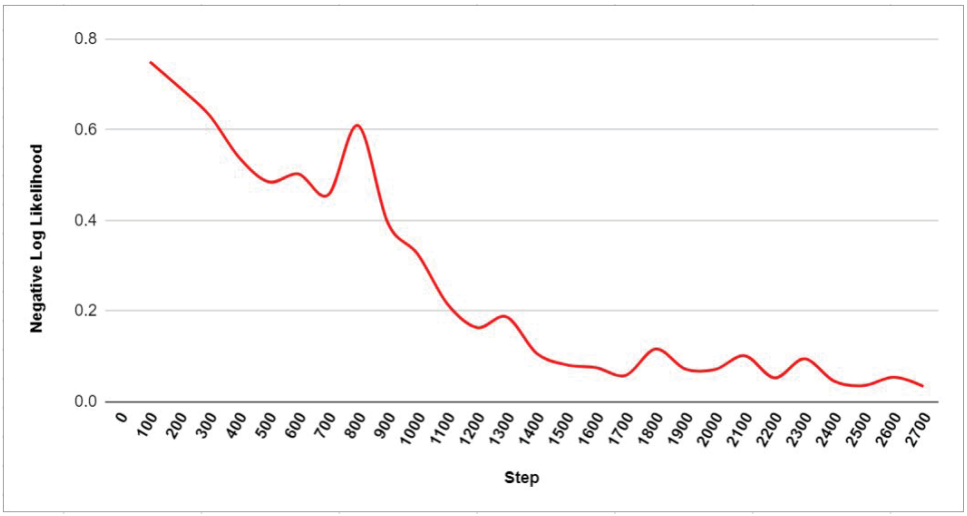
Four different sets of adversarial training executions were made using the mentioned pre trained weights with loss values mentioned in Table I. Initial adversarial training experiments were made using the obtained classifier with loss function value at 0.348. Training tended to start with higher NLL values than the latter experiments, and it dropped more slowly till reaching a minimum value around 1.25 just before exploding into values around 4.0, which represent failure. NLL evolution can be observed in Fig.8a. When testing generator inference capabilities around the saved weights around 1.25 we noticed that MIDI outputs were already beginning to show the desired characteristics. These first glimpse of jazz chord sequences can be heard in the mentioned results MIDI files repository.

[5]https://github.com/amazon-science/transformer-gan

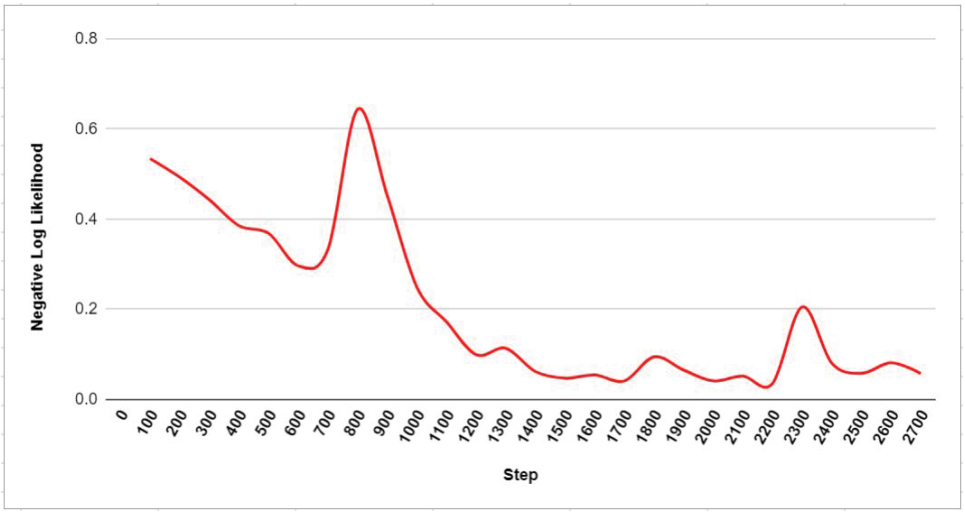
[6]https://drive.google.com/drive/folders/1ogtsvgWgwdL6vIXPNdvyDjEFSwlSaLm1?usp=share_link



(a) BERT classifier loss = .348.



(b) BERT classifier loss = .452.



(c) BERT classifier loss = .547.

Fig. 8. Evolution of NLL during adversarial training

Further adversarial training experiments using pre trained classifier weights with higher loss values showed more stable training yet chaotic, with the difference that NLL didn't explode towards high and undesired values, but showed a tendency to 0 over time. For this reason, adversarial training must be stopped before NLL reaches to the closer to zero range. When testing generator weights with desired NLL values we found out that the resulting data achieved our expectations, for the resulting data sounds very jazz oriented and presents coherence musically speaking. For progress evolution of NLL metric through adversarial training using classifiers at loss equal to 0.452 and 0.547, please refer to Fig. 8b and Fig. 8c respectively.

D. Subjective Evaluation

As seen in previous efforts to validate and evaluate generative models for symbolic music generation, we took the advice to perform complementary validation based in subjective evaluation through experts judgement using Likert scale as used in [19]. We asked a group of Jazz musicians to elaborate judgements over the generated chord sequences created by them through our proposed user interface.

Regarding to collected experts judgements and opinions summarized in Tab II, one important observation given by the experts is that generated sequences do not evidence a clear beginning and end. Voice leading is fluid but chords frequently do not go into most typical directions, which is valid in the jazz style context. It would be more idiomatic to get sequences that have an ending towards a tonic or root chord, for experts perceive that in some cases the sequences do not resolve in the best way. Artificially generated sequences expose a clear sense of direction and voice leading, which contain both typical and predictable jazz cadences and modern ones, which tend to be perceived as more unexpected and unpredictable. Experts pointed that some sequences exhibit pattern repetition and variations through them which result very aesthetic and enjoyable.

E. Discussion

A concrete validation of the proposed model also depends on the comparison carried out over other similar models performance ([8], [10] and [7]) to enable a clear way to compare coherence-length relationship between generated artificial data results. In each one of the related symbolic music generation research works, training was carried out much faster due to differences in capacity and computational resources. To have a glimpse of each of these works it was not necessary to execute their proposed models, rather was enough to observe their outputs in detail and analyze the data presented in their works. In the case of our work there is a clear difference in results, for other techniques don't expose generation of symbolic music limited to four or more notes per chord, in homophonic execution and with the reductionist arrangement of data to be found in our proposed dataset.

V. Conclusions

Experts considered that the proposed solution could be very useful as a starting point idea generator. The artificial sequential data could be later edited and incorporated into music arrangements organized and worked out by human operators.

Our obtained results showed that musicians could use the artificially generated data as starting point to work onto more complex compositions. This starting point could save composers time on thinking and brainstorming into the almost endless possibilities and paths when composing original music. ChordGAN could serve as a potent tool to increase the artist creativity and avoid to fall into seriously workflow stagnation conditions such as burnout, frustration or inability to propose different ideas and break with repetitive patterns.

As seen in other use cases using generative modelling [20], [21], [22], this symbolic music generator also shows particularities that reveal that a computer is creating the music, not a human: generated data constantly evades chord resolution. In less music wise terms, the sequences usually end without a proper closing of the musical expression.

This should be a point to develop on further research, for most musical expressions usually have a well defined beginning and ending, unless the composer wants to portray a more modernist, mysterious or experimental type of composition.

In future works, we aim to improve the model by adding tags to each chord during each training dataset sequence. This would enable inference capabilities to generate chord sequences with their common chord nomenclature expressions as seen in most Jazz lead sheets. This would permit musicians to keep track on upcoming chord executions, letting them to anticipate on upcoming chords to improvise melodic lines, bass lines or complementary chords in real time while performing live. Lastly, it will be very important to extend corpus volume to allow a richer pre training.

References

[1] S. Fürniss, "Aka polyphony: Music, theory, back and forth," in *Analytical Studies in World Music: Analytical Studies in World Music*. Oxford University Press, 2006.

[2] S. Pahaut and C. Meyer, "Une voix multiple: Entretien avec simha arom," *Cahiers de musiques traditionnelles*, vol. 6, p. 185, 1993.

[3] S. Klempe, "Implicit polyphony: A framework for understanding cultural complexity," *Culture & Psychology*, vol. 24, p. 1354067X1771639, 2017.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017, pp. 5998–6008.

[5] C. A. Huang, A. Vaswani, J. Uszkoreit, N. Shazeer, C. Hawthorne, A. M. Dai, M. D. Hoffman, and D. Eck, "An improved relative self-attention mechanism for transformer with application to music generation," *CoRR*, vol. abs/1809.04281, 2018.

[6] M. Zeng, X. Tan, R. Wang, Z. Ju, T. Qin, and T. Liu, "Musicbert: Symbolic music understanding with large-scale pre-training," in *ACL/IJCNLP (Findings)*, ser. Findings of ACL, vol. ACL/IJCNLP 2021, 2021, pp. 791–800.

[7] A. Muhamed, L. Li, X. Shi, S. Yaddanapudi, W. Chi, D. Jackson, R. Suresh, Z. C. Lipton, and A. J. Smola, "Symbolic music generation with transformer-gans," in *AAAI*. AAAI Press, 2021, pp. 408–417.

[8] G. Hadjeres, F. Pachet, and F. Nielsen, "Deepbach: a steerable model for bach chorales generation," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 70. PMLR, 2017, pp. 1362–1371.

TABLE II. ONLINE SURVEY ANSWERS TO SUBJECTIVELY EVALUATE A SET OF CHORD SEQUENCES GENERATED BY THE PROPOSED MODEL

Question	Likert Score
The generated harmonic sequences are coherent?	3.67
The generated harmonic sequences exhibit Jazz properties?	4.22
The chord sequences do not contain errors?	3.78

[9] A. Ranjan, V. N. J. Behera, and M. Reza, "Using a bi-directional LSTM model with attention mechanism trained on MIDI data for generating unique music," *CoRR*, vol. abs/2011.00773, 2020.

[10] C. A. Huang, A. Vaswani, J. Uszkoreit, I. Simon, C. Hawthorne, N. Shazeer, A. M. Dai, M. D. Hoffman, M. Dinculescu, and D. Eck, "Music transformer: Generating music with long-term structure," in *ICLR (Poster)*. OpenReview.net, 2019.

[11] J. Dyer and S. Sadie, "The new grove dictionary of music and musicians," *Speculum*, vol. 58, p. 528, 1983.

[12] L. Mihelac and J. Povh, "The impact of the complexity of harmony on the acceptability of music," *ACM Trans. Appl. Percept.*, vol. 17, no. 1, pp. 3:1–3:27, 2020.

[13] P. O'Grady, "The analogue divide: interpreting attitudes towards recording media in pop music practice," *Continuum*, vol. 33, pp. 446–459, 2019.

[14] M. Abbasi, B. P. Santos, T. Pereira, R. Sofia, N. R. C. Monteiro, C. J. V. Simões, R. Brito, B. Ribeiro, J. L. Oliveira, and J. P. Arrais, "Designing optimized drug candidates with generative adversarial network," *J. Cheminformatics*, vol. 14, no. 1, p. 40, 2022.

[15] Z. Dai, Z. Yang, Y. Yang, J. G. Carbonell, Q. V. Le, and R. Salakhutdinov, "Transformer-xl: Attentive language models beyond a fixed-length context," in *ACL*, 2019, pp. 2978–2988.

[16] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *NAACL-HLT (1)*. Association for Computational Linguistics, 2019, pp. 4171–4186.

[17] J. Wu, J. Tang, J. Zhang, and J. Di, "Coherent noise suppression in digital holographic microscopy based on label-free deep learning," *Frontiers in Physics*, vol. 10, p. 880403, 2022.

[18] H. L. Corp, *The Real Book - Volume I - Sixth Edition - Mini Edition: C Edition*. Hal Leonard Publishing Corporation, 2007.

[19] C. Hernandez-Olivan, J. A. Puyuelo, and J. R. Beltrán, "Subjective evaluation of deep learning models for symbolic music composition," *CoRR*, vol. abs/2203.14641, 2022.

[20] L. Cornejo, R. Urbano, and W. Ugarte, "Mobile application for controlling a healthy diet in peru using image recognition," in *FRUCT*. IEEE, 2021, pp. 32–41.

[21] E. Burga-Gutierrez, B. Vasquez-Chauca, and W. Ugarte, "Comparative analysis of question answering models for HRI tasks with NAO in spanish," in *SIMBig*, vol. 1410. Springer, 2020, pp. 3–17.

[22] D. J. Lozano-Mejía, E. P. Vega-Uribe, and W. Ugarte, "Content-based image classification for sheet music books recognition," in *2020 IEEE EirCON*, 2020, pp. 1–4.