ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF ARTS AND SOCIAL SCIENCES

AN APPLICATION OF NEURAL NETWORK-BASED MUSIC GENERATION MODELS IN THE CONTEXT OF MODERN AND CONTEMPORARY MUSIC

M.A. THESIS

Tuğrul Orkun AKYOL

Department of Music

Music Programme

DECEMBER 2019

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF ARTS AND SOCIAL SCIENCES

AN APPLICATION OF NEURAL NETWORK-BASED MUSIC GENERATION MODELS IN THE CONTEXT OF MODERN AND CONTEMPORARY MUSIC

M.A. THESIS

Tuğrul Orkun AKYOL
(409171198)

Department of Music

Music Programme

Thesis Advisor: Asst. Prof. Dr. Emmanouil EKMEKTSOGLOU

DECEMBER 2019

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ SOSYAL BİLİMLER ENSTİTÜSÜ

NÖRAL AĞ BAZLI MÜZİK JENERASYON MODELLERİNİN MODERN VE ÇAĞDAŞ MÜZİK BAĞLAMINDA UYGULAMASI

YÜKSEK LİSANS TEZİ

Tuğrul Orkun AKYOL
(409171198)

Müzik Anabilim Dalı

Müzik Programı

Tez Danışmanı: Dr. Öğr. Üyesi Emmanouil EKMEKTSOGLOU

ARALIK 2019

Tuğrul Orkun AKYOL, a M.A. student of ITU Graduate School of Arts and Social Sciences student ID 409171198, successfully defended the thesis entitled "AN APPLICATION OF NEURAL NETWORK-BASED MUSIC GENERATION MODELS IN THE CONTEXT OF MODERN AND CONTEMPORARY MUSIC", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor : **Asst. Prof. Dr. Emmanouil EKMEKTSOGLOU**.....
Istanbul Technical University

Jury Members : **Assoc. Prof. Dr. Jerfi AJİ** .....
Istanbul Technical University

Assoc. Prof. Dr. Tolga Zafer ÖZDEMİR .....
Bilgi University

Date of Submission : 15 November 2019
Date of Defense : 9 December 2019

FOREWORD

First of all I would like to thank my composition instructors. Dr. Reuben de Lautour, Dr. Pieter Snapper and Dr. Jeremy Woodruff helped me develop into the musician and scholar I am today.

I am grateful to Assoc. Prof. Dr. Jerfi Aji, who lent a hand in every problem I had throughout my studies and supported me during this research.

Lastly I appreciate the guidance of my advisor Dr. Emmanouil Ekmektsoglou and Dr. Konstantinos Vasilakos; without them this thesis would not have been possible.

December 2019 Tuğrul Orkun AKYOL

TABLE OF CONTENTS

Page

FOREWORD ..... **vii**
TABLE OF CONTENTS ..... **ix**
ABBREVIATONS ..... **xi**
LIST OF EXAMPLES ..... **xiii**
LIST OF FIGURES ..... **xv**
SUMMARY ..... **xvii**
ÖZET ..... **xix**
1. INTRODUCTION **1**
1.1 Purpose 1
1.2 Neural Networks 2
1.2.1 Working principles 2
1.2.2 Brief history 4
1.3 Literature Review 5
2. MODELS **9**
2.1 Onsets and Frames 9
2.2 Performance RNN 11
2.2.1 First experiment 14
2.2.1.1 Music of Messiaen 14
2.2.1.2 Analysis of outputs 17
2.2.2 Second experiment 24
2.2.2.1 Hyperparameters 24
2.2.2.2 Analysis of outputs 25
2.2.3 Third experiment 31
2.2.3.1 Music SOM 31
2.2.3.2 Analysis of outputs 34
2.2.4 Fourth experiment 39
2.2.4.1 Improvisational framework 39
2.2.4.2 Compositional thinking 40
3. CONCLUSIONS AND DISCUSSION **47**
REFERENCES **49**
APPENDICES **53**
APPENDIX A: Commands for running Onsets and Frames locally 54
APPENDIX B: First two pages of the longer output of the first experiment 55
APPENDIX C: Code fragment of Performance RNN for the Performance with the 'performance_with_dynamics_compact' configuration 57
APPENDIX D: First two pages of the longer output of the second experiment 58
APPENDIX E: Calls for the training function of Music SOM 60
APPENDIX F: First two pages of the longer output of the third experiment 61
APPENDIX G: Sample of improvisation transcription 63
APPENDIX H: First two pages of the longer output of the fourth experiment 65
APPENDIX I: Comprovisational miniature 67
CURRICULUM VITAE **71**

ABBREVIATONS

ANN **:** Artificial neural network

BPM **:** Beats per minute

GPU **:** Graphic processing unit

LSTM **:** Long short-term memory

MAPS **:** MIDI aligned piano sounds

MFCC **:** Mel frequency cepstral coefficients

MIDI **:** Musical instrument digital interface

XML **:** Extensive markup language

PDF **:** Portable document format

RBM **:** Restricted Boltzmann machine

RNN **:** Recurrent neural network

SOM **:** Self organizing maps

LIST OF FIGURES

Page

Figure 1.1 : Architecture of a fully connected feedforward ANN **3**
Figure 1.2 : Biological inspiration of a perceptron..... **4**
Figure 2.1 : First two bars of *Le Chocard des Alpes* **10**
Figure 2.2 : Transcription of the first two bars of *Le Chocard des Alpes*..... **10**
Figure 2.3 : Excerpt from a sample output of the Colab notebook of Music Transformer..... **14**
Figure 2.4 : Chromatic cells mentioned in Messiaen (1956) **15**
Figure 2.5 : Messiaen's 3rd mode of limited transposition..... **16**
Figure 2.6 : Messiaen's 4th mode of limited transposition..... **16**
Figure 2.7 : Messiaen's 6th mode of limited transposition..... **16**
Figure 2.8 : Dominant chord and the chord of resonance..... **17**
Figure 2.9 : Bars 4-6 of the first 30 second-output, first experiment **17**
Figure 2.10 : Bars 13-14 of the first 30 second-output, first experiment **18**
Figure 2.11 : Bars 4-5 of the fourth 30 second-output, first experiment **18**
Figure 2.12 : Bars 13 of the fourth 30 second-output, first experiment **19**
Figure 2.13 : Bars 5-6 of the fifth 30 second-output, first experiment..... **19**
Figure 2.14 : Velocity data of figure 2.13..... **20**
Figure 2.15 : Chromatic cells in the 5 minute-output, first experiment **20**
Figure 2.16 : Repeated motive in the 5 minute-output, first experiment..... **20**
Figure 2.17 : Reiteration of the root motion in the 5 minute-output, first experiment..... **21**
Figure 2.18 : Third mode code and F minor material in the 5 minute-output, first experiment **22**
Figure 2.19 : Bars 89-90 of the 5 minute-output, first experiment..... **22**
Figure 2.20 : Bars 19-23 of Messiaen's *Catalogue d'Oiseaux*, no.7..... **23**
Figure 2.21 : Training accuracy of the first experiment **23**
Figure 2.22 : Training loss of the first experiment..... **24**
Figure 2.23 : Bars 15-16 of the second 1 minute-output, second experiment **25**
Figure 2.24 : Bar 4 of the third 1 minute-output, second experiment **26**
Figure 2.25 : Bars 29-30 of the third 1 minute-output, second experiment..... **26**
Figure 2.26 : Bars 2-3 of the 30 second-output, second experiment..... **26**
Figure 2.27 : Bars 11-12 of the 30 second-output, second experiment **27**
Figure 2.28 : Bars 35-36 of the 5 minute-output, second experiment **27**
Figure 2.29 : Bar 58 of the 5 minute-output, second experiment..... **28**
Figure 2.30 : Bars 94-96 of the 5 minute-output, second experiment **28**
Figure 2.31 : Special chords in bars 64 and 138 of the 5 minute-output, second experiment..... **28**
Figure 2.32 : Lullabyish musical idea in bars 114-118 of the 5 minute-output, second experiment **29**

Figure 2.33 : Training accuracy of the training (orange) and validation (blue) set, second experiment **30**
Figure 2.34 : Training loss of the training (orange) and validation (blue) set, second experiment **30**
Figure 2.35 : Map from the first experiment with Music SOM **34**
Figure 2.36 : Bars 2-4 of the first 30 second-output, third experiment **35**
Figure 2.37 : Bars 13-14 of the second 30 second-output, third experiment **35**
Figure 2.38 : Bar 16 of *Les Travaux et les Jours, no.1* **35**
Figure 2.39 : Bars 4-6 of the third 30 second-output, third experiment **36**
Figure 2.40 : First bar of *Klavierstück IX* **36**
Figure 2.41 : Bars 25-26 of the 5 minute-output, third experiment **37**
Figure 2.42 : Bars 129-130 of the 5 minute-output, third experiment **37**
Figure 2.43 : Transpositionally equivalent pitch class sets in the 5 minute-output, third experiment **37**
Figure 2.44 : Bar 141 of the 5 minute-output, third experiment **38**
Figure 2.45 : Training accuracy of the training (orange) and validation (blue) set, third experiment **38**
Figure 2.46 : Training loss of the training (orange) and validation (blue) set, third experiment **39**
Figure 2.47 : Spectra stretched with scale factors 0.8, 0.9 and 1.2, respectively **40**
Figure 2.48 : Sample output containing triadic material **41**
Figure 2.49 : Sample output containing a single-voice run **41**
Figure 2.50 : Training accuracy of the training (orange) and validation (blue) set, fourth experiment **41**
Figure 2.51 : Training loss of the training (orange) and validation (blue) set, fourth experiment **42**
Figure 2.52 : Bars 12-14 in the fifteenth 1 minute-output, fourth experiment **43**
Figure 2.53 : Bars 3-9 in the fifteenth 1 minute-output, fourth experiment **44**
Figure 2.54 : Bars 8-14 in the thirteenth 1 minute-output, fourth experiment **44**
Figure 2.55 : Bars 2-8 in the seventh 1 minute-output, fourth experiment **45**
Figure A.1 : Original code for running Onsets and Frames locally **54**
Figure A.2 : Corrected code for running Onsets and Frames locally **54**
Figure A.3 : Sample output from the first experiment **55**
Figure A.4 : 'performance_with_dynamics_compact' configuration **57**
Figure A.5 : Sample output from the second experiment **58**
Figure A.6 : Original code for calling the training function of Music SOM **60**
Figure A.7 : Updated code for calling the training function of Music SOM **60**
Figure A.8 : Sample output from the third experiment **61**
Figure A.9 : Sample of improvisation transcription **63**
Figure A.10 : Sample output from the fourth experiment **65**
Figure A.11 : Comprovisational miniature **67**

LIST OF TABLES

Page

Table 2.1 : The number of wins in cells won by Messiaen more than 200 times **.....33**

AN APPLICATION OF NEURAL NETWORK-BASED MUSIC GENERATION MODELS IN THE CONTEXT OF MODERN AND CONTEMPORARY MUSIC

SUMMARY

The purpose of this thesis is to test the performance of the artificial neural network-based music generation models in a different musical context and to determine if they can be used as composition assistance tools by the contemporary composer. A brief explanation of the working principles and recent history of ANNs are given before moving onto the conducted experiments.

For the purposes of the thesis, a model called Performance RNN, developed by Project Magenta, is trained three times with different datasets and/or hyperparameters. The model is capable of creating performances with feeling: it is sensitive to time intervals as small as eight milliseconds and it can process dynamic information. The datasets are in MIDI format and they are compiled from audio files with the help of another neural network model called Onsets and Frames, which transcribes raw audio files of piano music into MIDI files.

For the first experiment a dataset of Messiaen's complete piano works is used, which spans over six hours. Messiaen is chosen because of his relative consistent musical language and large output of solo piano works. The configuration used takes dynamics into account. Several features reminiscent of Messiaen can be observed in the outputs, however, the training accuracy is as low as forty percent. The model performs well in terms of pitch content but struggles with rhythmic and formal structures.

The same dataset is used for the second experiment, but this time the dynamics are eliminated and a small part of the dataset is set aside for validation. Also, the model is trained with an additional layer to increase its learning capacity. There is a slight improvement in accuracy of the test set, but the model does not do well on the validation test: it can not generalize over musical features of Messiaen's works. The outputs are rhythmically more chaotic and no drastic improvements are observed in the pitch content.

For the third experiment the dataset is expanded with works by other composers. These composers are chosen with the help of a neural network model called Self Organizing Maps, which reduces the dimension of data and displays the similarities among them. In this case, the data is mostly spectral features extracted from audio files. In the end an hour of Stockhausen, Schönberg, Murail and Ferneyhough's piano music is added to the dataset. The model reacts to the addition of the works of these composers; various musical examples are given to address this reaction.

The last experiment is run with a dataset of improvisations made by the author of the thesis. The outputs of this experiment are then used to compose a piano miniature. Compositional context is widely discussed in the related section.

In the last section the importance of the size and content of the chosen dataset for these experiments is underlined and the difficulties of compiling a dataset of

contemporary music is discussed. Finally, the impact made by the used transcription model and the encoding type used in the generation model is briefly mentioned.

NÖRAL AĞ BAZLI MÜZİK JENERASYON MODELLERİNİN MODERN VE ÇAĞDAŞ MÜZİK BAĞLAMINDA UYGULAMASI

ÖZET

Bu tezin amacı nöral ağ bazlı müzik jenerasyon modellerini farklı bir müzikal bağlamda test etmek ve bu modellerin çağdaş besteciler tarafından kompozisyona yardımcı bir araç olarak kullanılıp kullanılmayacağını anlamaktır. Yapılan deneylerden önce nöral ağların çalışma prensipleri ve tarihleri ile ilgili kısaca bilgi verilmiştir.

Bu doğrultuda Project Magenta tarafından geliştirilen Performance RNN isimli bir model değişik dataset ve/ya da hiperparametreler ile üç kez eğitilmiştir. Söz konusu model dinamik bilgileri dikkate aldığından ve sekiz milisaniyeye kadar zaman aralıklarına duyarlı olduğundan hissiyat içeren performanslar yaratabilmektedir. Modeli eğitmek için kullanılan datasetteki dosyalar MIDI formatındadır ve bu formata ses dosyalarından Onsets and Frames isimli başka bir nöral ağ tarafından dönüştürülmüştür.

Birinci deney için Messiaen'in bütün piyano eserleri kullanılmıştır. Bu eserler yaklaşık altı saat sürmektedir. Messiaen yazdığı müziklerde nispeten tutarlı bir dil kullanması ve geniş bir piyano repertuarı olması nedeniyle seçilmiştir. Modelin ilk deneyde kullanılan konfigürasyonu dinamik bilgileri dikkate almaktadır. Bu deney sonucunda elde edilen müziklerde Messiaen'in müziğini andıran özellikler bulunsa da model eğitiminin doğruluk derecesi yüzde 40 gibi düşük bir rakamda kalmıştır. Modelin nota içeriği olarak başarılı müzikler ürettiğini söylenebilse de aynı şeyi ritmik içerik ve formal yapı olarak söylemek mümkün değildir.

Aynı dataset ikinci deney için de kullanılmış, fakat bu kez modelden dinamik içerik elenmiş ve datasetin küçük bir kısmı modeli doğrulamak üzere kenara ayrılmıştır. Ayrıca modelin öğrenme kapasitesine artırabilmek için nöral ağa ekstra bir katman eklenmiştir. Bu değişikliklerin sonucunda eğitim setinin doğruluk derecesinde küçük bir artış görülse de model doğrulama setinde iyi bir performans gösterememiştir. Bundan yola çıkarak modelin Messiaen'in müziklerini genelleyebilecek kadar iyi öğrenemediği söylenebilir. Bu deneyin ürettiği müziklere bakıldığında ritmik olarak daha kaotik yapılar görülmektedir. Nota içeriği olarak ise ilk deneye kıyasla ciddi bir gelişme gözlenmemiştir.

Üçüncü deneyde datasete değişik bestecilerden çalışmalar eklenmiştir. Bu besteciler Self Organizing Map isimli bir nöral ağ modelinin yardımıyla seçilmiştir. Bu model yüksek boyutlu verilerin boyutunu düşürüp verileri aralarındaki benzerliklere göre görselleştirebilmektedir. Bu deneyde veri olarak ses dosyalarından spektral nitelikler elde edilmiştir. Bu sürecin sonunda her biri birer saat olmak üzere Stockhausen, Schönberg, Murail ve Ferneyhough'un piyano müzikleri datasetine eklenmiştir. Eğitim sonrası üretilen müziklere bakarak modelin bu eklemelere tepki verdiği söylenebilir. Bu konu ilgili bölümde birçok müzikal örnek üzerinden tartışılmıştır.

Son deneyin dataseti yazarın kendi doğaçlamalarından oluşmaktadır. Bu deneyde elde edilen çıktılar bir piyano miniyatürünün bestelenmesinde kullanılmıştır. İlgili bölümde çıktıların kompozisyon bağlamında kullanımı detaylı bir biçimde ele alınmıştır.

En son bölümde bu tarz çalışmalar için seçilen datasetin boyut ve içeriğinin önemi belirtilmiş, çağdaş müzik bağlamında verip toplamanın zorluğu tartışılmıştır. Son olarak, ses dosyalarının MIDI formatına transkripsiyonunu yapan modelin ve üretici modelde kullanılan kodlama biçiminin önemine kısaca değinilmiştir.

1. INTRODUCTION

1.1 Purpose

Artificial neural networks (ANN) have a relatively long history. It was as early as 1943 when McCulloch-Pitts Neuron was introduced, followed by Perceptron in 1957 – the two primary ancestors of today's models. However, up until recently, the challenges in effectively training multi-layer networks have limited the interest in ANNs. This situation changed when a pre-training technique was proposed by Hinton et al. (2006) and a deep neural network algorithm, i.e., AlexNet won the ImageNet Large Scale Visual Recognition Contest using this technique in 2012 (Briot et al., 2019). Music generation research also joined the trend, a successful demonstration of which has been Google AI's launch of Magenta research project which focuses on machine learning applications in arts. Despite the broad definition, almost all demos on their website use various architectures of ANNs.

If an algorithm is very generally defined as a formalizable and abstractable procedure (Nierhaus, 2009), then we can say that numerous composers were assisted by algorithmic procedures. Famous examples include Mozart's dice game, which permutates pre-composed measures of music according to the outcomes of dice rolls. A more recent example from the 20th century is Xenaxis, who utilized stochastic models for music composition. Magenta project tends to a similar purpose. In their blog post about Magenta Studio they state their aim as developing models specifically targeted to the goals of creators and then developing easy-to-use tools based on these models (Roberts et al., 2019). However, most of the ANN models are trained on MIDI datasets of common practice period music, which is of limited use to a contemporary music composer. ANNs are able to generate music based on the corpus style on which they have been trained, but once they are trained, only a few, if any, interventions on the output may be made by humans.

In this context the primary purpose of this research is to test the performance of a recent ANN model of Magenta called Performance RNN in the context of modern and

contemporary music, which can be more sophisticated than the music composed in the tonal idiom in the common practice period. The second aim is to investigate whether these models are useful as composition assistance tools for the contemporary composer. To address these aims, I compile my own datasets and generate polyphonic music by training the ANN on them.

The title of the thesis includes labels of both modern and contemporary music for the lack of a better term. Modern music refers to the departures in musical language that occurred around the turn of twentieth century (Metzer, 2011). The term generally does not imply a consistent musical style but implies a break from tradition and focus on innovation. Composers of the Second Viennese School, Messiaen and Stockhausen used in the datasets of this thesis can be classified under this label. On the other hand, I used the term contemporary music in a literal sense, as the music being written today. This refers to the use of Murail and Ferneyhough in the datasets and more importantly to my own improvisation and compositions. The common denominator of the composers classified under both labels is that they do not use a tonal language in their works.

1.2 Neural Networks

Prior to introducing the literature review on deep learning in music, I would like to take a moment to briefly introduce the working principles and the history of ANNs. Although a comprehensive review of the mathematical framework of ANNs are not in scope of this thesis, some insights may prove to be beneficial for those with a music background but are inexperienced in the field of machine learning.

1.2.1 Working principles

ANNs are mathematical models which loosely imitate the signal flow of the human nerve cells. Their main capability is to extract and replicate features and patterns in a dataset, rendering them useful for various classification, clustering, prediction and generation tasks. Essentially, ANNs are composed of multiple layers, each one containing a certain number of neurons. The first layer is called the input layer, the last one the output layer and the layers inbetween are called the hidden layers. In a fully connected feedforward network, each neuron in a layer is connected to all the other neurons in the preceding and succeeding layers. There are various architectures

differing in significant ways, but feedforward networks have arguably the simplest form, which makes them particularly useful for getting acquainted with ANNs. Figure 1.1 depicts a small scale example of this architecture with circles representing individual neurons.

Each neuron in the input layer accepts a number as input. The arcs connecting the neurons have weights associated with them. As data flows through the network, the inputs are multiplied with these weights and these multiplications are summed up before being fed into the non-linear activation function of the next neuron, eventually reaching the output layer.

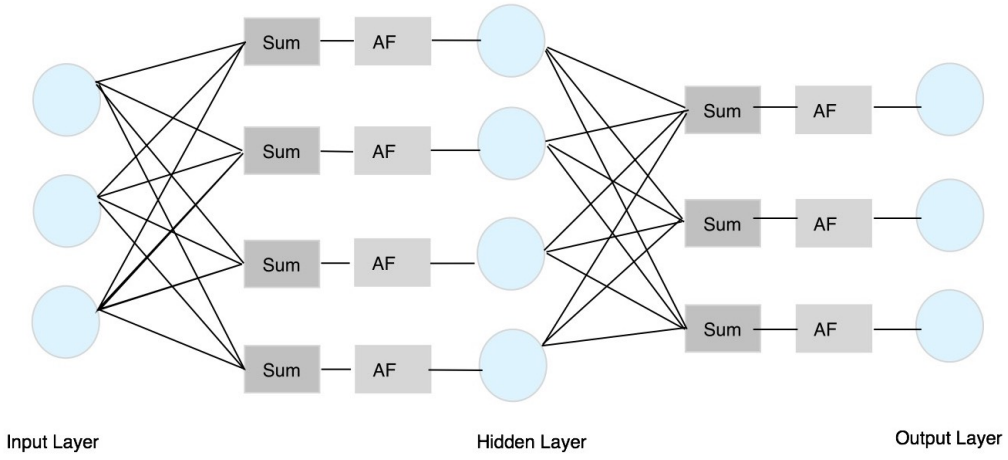


Figure 1.1: Architecture of a fully connected feedforward ANN[1]

While in every ANN the weights are initially random, it is possible to train the network and to update these weights in order to obtain the output we desire. In musical terms, this can simply be the next note given a previous sequence, encoded in numerical values. Performance RNN on the other hand has a much more sophisticated way of encoding musical material as it generates polyphonic music. I will talk about encoding alternatives in the Literature Review section (1.3) and as I go over Performance RNN in the next chapter (2.2).

[1] Illustrated based on the figure in Briot et al (2019, p.53).

To be able to train a network, we first need a measure of fitness. That is what a loss function is for: measuring the deviation from the desired output after the evaluation of each batch of input by the ANN. Next step is to link this loss back to the weights between pairs of neurons. This process utilizes the backpropagation algorithm to compute the gradient (vector of partial derivatives) via the chain rule, which allows the information from the loss function to flow backwards through the network (Goodfellow et al., 2016). Gradient descent algorithm completes the procedure by updating the weights in the opposite direction of the gradient, which guarantees to reduce loss[2]. The amount of the movement along the opposite direction of the gradient depends on the learning rate, which is a very small positive number fixed beforehand according to the collective experience of machine learning researchers.

1.2.2 Brief history

In their seminal article, McCulloch and Pitts (1943) propose the first model of a neuron. This model is confined to learn boolean functions, as all its inputs and outputs are 0 or 1 (Chandra, 2018a). Perceptron, proposed by Rosenblatt (1958) built upon this and introduced numerical weights for inputs and a method for learning them, hence the inputs of the Perceptron do not have to be boolean values (Chandra, 2018b). Perceptron was criticized by Minsky and Papert (1969) primarily because it could not learn nonlinearly seperable functions, resulting in a period where AI research stagnated for more than a decade (Kurenkov, 2015).

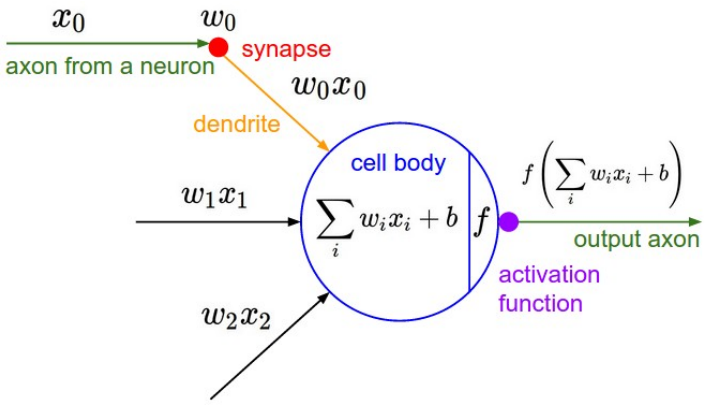


Figure 1.2: Biological inspiration of a perceptron[3]

[2] For the mathematical proof, see Nielsen (2019).
[3] Retrieved from http://cs231n.github.io/neural-networks-1/ on 04.11.2019.

The period of stagnation ended when Rumelhart et al. (1986) proposed the application of backpropagation algorithm on neural nets and addressed Minsky and Papert criticisms. Implementations of various architectures and applications to real world problems followed in the next few years (Kurenkov, 2015).

One such architecture is the Recurrent Neural Networks (RNN) that are able to process and output sequential data, which is crucial for musical applications. They may have connections between neurons within the same layer and to a neuron from itself in the form of a loop - creating a memory structure within the network. Training of RNNs suffered from short term memory before the Long Short-Term Memory (LSTM) idea was proposed by Schmidhuber and Hochreiter (1997). LSTMs are special kind of RNNs; each LSTM cell keeps track of a cell state in addition to its output. The cell state is controlled by various gates which are neural nets themselves (Nguyen, 2018). This way the memory of the cells is improved to keep the more relative information.

ANNs became increasingly popular with the pre-training technique of Hinton (2006) involving non-random initialization of weights, the convincing win of AlexNet in an image recognition competition in 2012 and the availability of general purpose Graphical Processing Units (GPU). The popularity of RNNs in particular owe to Karpathy's viral post "The Unreasonable Effective of Recurrent Neural Networks" (2015), where he showed that a simple RNN can recreate the look and feel of any text (McDonald, 2017).

1.3 Literature Review

Last thirty years, the last decade in particular witnessed many examples of music generation with deep learning methods. In this section, I will try to introduce some of the most significant attempts[4].

The first instance of music generation with ANNs is achieved by Todd (1989). He used an RNN structure for generating monophonic melodies. The melodies of the training set are transposed to C major beforehand. Every output of the network represents a single pitch and each input vector represents a single time step. In 2002, Eck moved this experiment to an LSTM framework in the context of blues improvisation. Eck's

[4] For a more in-depth literature survey, see Briot et al. (2019).

work does not impose a key restriction and the model is able to learn the harmonic relationship between pitches for itself.

Another task for ANN models besides monophonic music generation is harmonization. HARMONET, developed by Hild et al. (1992), harmonizes melodies in the style of J.S. Bach. It is a hybrid system consisting of three parts: (1) An RNN responsible for deriving a harmonic skeleton from the melody with pitches encoded as the set of harmonic functions that contains them; (2) A symbolic algorithm generating the chord skeleton out of the harmonic skeleton; and (3) A neural net inserting eighth note ornamentations to chords. Another ANN architecture called Restricted Boltzman Machine (RBM) is used by Boulanger-Lewandowski et al (2012), which generates chord progressions based on Bach chorales. Prior to training, the chorales are transposed to the key of C major or C minor. The representation used is called a multi-hot encoding, in which the input vector has the size of the number of available pitches and every pitch is represented by a boolean value: the corresponding element of the vector is 1 if the pitch is being played at any time step. A more recent system called DeepBach (Hadjeres et al., 2017) generates chorale harmonizations in a more sophisticated manner. It uses two LSTM networks summing up the past and future information, a feedforward network responsible for the current notes and a second one merging all the output generated by the previous networks. The choice of representation is a multi-one-hot encoding: in each voice just a single pitch can be played at any time. (Briot et al., 2019).

Moving on to another task of polyphonic music generation, Boulanger-Lewandowski et al. (2015) associated an RNN to the RBM structure mentioned above. In this hybrid structure RNN models the temporal sequence and the RBM models the pitches that should be played together (Briot et al., 2019). The model is trained on four different datasets: J.S. Bach chorales, classical piano pieces, orchestral classical music and folk tunes. A similar model is also built by Johnson (2015) where he used a biaxial RNN with the first and second parts recurrent in time and notes, respectively. The model is trained on the MIDI files on Classical Piano Midi Page[5], which consists of common practice period piano works. A variation of Johnson's model is proposed by Mao et al (2018), which they named the DeepJ. This system focuses on the consistency of style,

[5] http://www.piano-midi.de/

as it takes an input vector of 23 numbers adding up to 1, corresponding to the 23 composers of common practice period that it was trained on. This way the users can opt for different percentage values of the styles of different composers in the output. This kind of style encoding is criticized as being to simplistic by Briot et al (2019) on the grounds that musical styles are not orthogonal to each other and share many characteristics.

One of the latest developments in the polyphonic music generation research is the Music Transformer[6] model developed by Huang et al. (2018) within the Magenta project. Transformer models are solely based on attention mechanisms, which allows for modeling dependencies without regard to their distance in the input or output sequences, as opposed to the sequential nature of recurrent models (Vaswani et al., 2017). With the help of this architecture, Music Transformer performs better than Performance RNN in terms of long term coherence. Samples for comparison can be found in the blog post about the model[7].

These examples of ANNs primarily operate on symbolic level but there is also research on music generation on the raw audio level – the drawback being the greater computational need for training the models and generating outputs. One example like this is the WaveNet system (van den Oord et al., 2016), which uses convolutional neural nets (frequently used in the visual domain) to create speech from text and to generate music. Convolutional networks are neural networks that use the mathematical operation of convolution in at least one of their layers. They are suitable for data with a grid-like topology (Goodfellow et al., 2016). In the case of audio files, temporality of the samples are represented in a one-dimensional grid.

[6] The model is not considered for this thesis as the code is not yet released.
[7] https://magenta.tensorflow.org/music-transformer

2. MODELS

2.1 Onsets and Frames

The first challenge in applying a machine learning tool to music is to compile a dataset. Performance RNN preprocesses and trains on MIDI data, however available MIDI files of compositions of 20th century and contemporary composers are scarce. Creating MIDI files manually from scratch would be very time consuming and therefore not feasible. Two alternative solutions are: (1) creating MIDI files from scores with an optical music recognition software; (2) using machine learning transcription models like Onsets and Frames of the Magenta Project.

The alternative of using an optical music recognition software requires the availability of PDF files of scores for every composition in the dataset. However this is not always the case. Obtaining hard copies of scores and scan them manually is time consuming as well as logistically and financially very difficult, if not impossible. Also, the quality of the resultant MIDI files decreases with the decreasing quality of PDF files. Frequent errors include inaccurate durations of single notes, resulting in incomplete measures. This is a must-fix error and it is only manually fixable. Due to all this factors second alternative is used for the purposes of the thesis.

The machine learning transcription model used in this thesis, i.e., Onsets and Frames, which was also used for creating the dataset for training the interactive Colaboratory version[8] of Music Transformer (Huang et al., 2018), is a deep learning model for transcribing polyphonic piano music employing deep recurrent and convolutional networks (Hawthorne et al., 2018). The model is able to extract the onsets, durations and velocity of notes from raw audio files and output MIDI files as a result. It outperformed similar models on two different test sets used by the authors. The sustain pedal changes are managed by extending the duration of the notes until the pedal-off message while preprocessing the MIDI files in the training set.

[8] https://colab.research.google.com/notebooks/magenta/piano_transformer/piano_transformer.ipynb

The first drawback for using the model is that we are restricted to piano music only. The second and more important one is the quality of MIDI transcriptions. The model is trained on the MAPS dataset compiled by Emiya et al. (2010) and the dataset contains music pieces aside from monophonic sounds and random chords. The music pieces used in the dataset are from the Classical Piano Midi Page containing a repertoire from the common practice period. Intuitively this should determine the style of music the model has a better performance with. For the sake of comparison, figures 2.1 and 2.2 show the first two bars of Messiaen's *Le Chocard des Alpes* from *Catalogue des Oiseaux* and its transcription. The MIDI file is quantized and converted to a score by Logic Pro X. Logic outputs measures of four quarter notes by default. Sibelius 7.5 is used for editing the layout for the screenshots, via the MusicXML files exported from Logic.

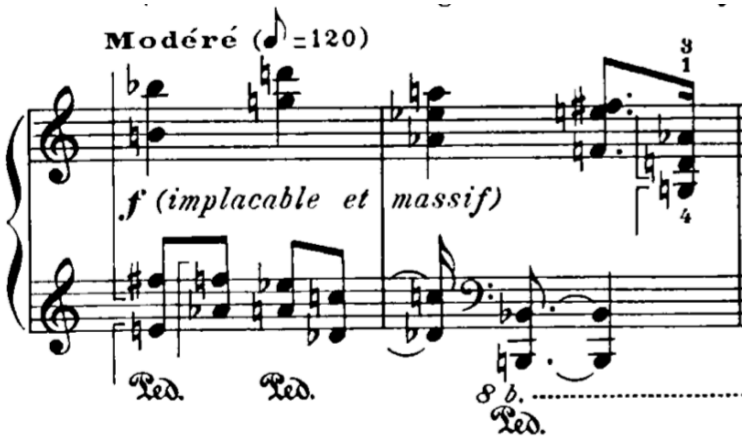


Figure 2.1: First two bars of *Le Chocard des Alpes*

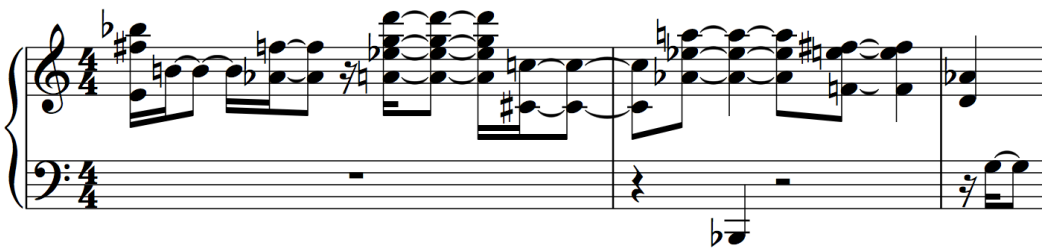


Figure 2.2: Transcription of the first two bars of *Le Chocard des Alpes*

Firstly we observe that the beat is converted to a half note in the transcription from the quarter note in the original score. The rhytmic values of the notes are far from accurate. Some of the chords are perceived as broken and some of the note onsets are late. This has to do with the fact that the model outputs MIDI with millisecond

durations based on the original performances, which is usually far from mechanical and can deviate from the score. Then it is quantized by Logic, which increases the error margin. The incapabilities of Logic are threefold: (1) The transcription does not hold notes properly when it detects the beginning of a new pitch, even though in the MIDI data notes are sustained for much longer than seen in the transcribed score – accounting for the sustain pedal as well[9]; (2) Logic does not recognize any other tuplets than the triplet; and (3) It does not generate particularly legible scores when dealing with extreme registers.

Returning to the performance of the model, we can see that it does incredibly well in the pitch domain. It only missed a single pitch in this case, the low B natural in the second bar. It was the success of the model in terms of pitch content that convinced me to use the model in this thesis. Having this observation in mind, I prioritized pitch content over rhythmical content when analyzing the outputs.

I ran the code for Onsets and Frames locally on my personal computer using mostly Youtube recordings of performances as audio sources. There is also an online interface of the model called Piano Scribe[10], but I had difficulties running it because of the upload speed limitations of my internet service provider. Moreover, for the short segments that I managed to upload, it produced worse results.

Lastly, it is also worth mentioning that by November 2019 the command published for running Onsets and Frames locally contains simple mistakes. The original and corrected version of the command can be found in Appendix A.

2.2 Performance RNN

Performance RNN is a neural network-based generative music model from the paper Oore et al. (2018), which is capable of learning expressive dynamics and timing and generating performance-like MIDI outputs. It uses an LSTM architecture with three layers of 512 neurons. The encoding of musical information is done by one-hot vectors with a dimension of 413, consisting of 128 note-on, 128 note-off, 125 time-shift and 32 velocity events, which correspond to the range of 128 notes, 125 time steps of

[9] Exact durations are shown if MIDI files are opened with Sibelius directly, but then the scores become illegible.
[10] https://piano-scribe.glitch.me/

multiples of 8 milliseconds (adding up to a second at most) and the whole range of velocity values divided into 32 bins. To clarify the encoding further, this means that a note is sustained after a note-on message until an input vector with the note-off message in the corresponding note is fed to the network. In between there may be vectors with time-shift messages and vectors indicating that other notes should be played. Input vectors with velocity messages are fed to the network consecutively to the vectors with note-on messages. The minimum time step of 8 milliseconds is quite sensitive and it contributes significantly to the expressiveness of the output.

As stated in Nierhaus (2009) and repeated in Oore et al. (2018), long term coherence can not be captured with ANN models properly. That is why formal considerations were not prioritized in the analysis of the outputs in the next sections. Music Transformer from Huang et al. (2018) seemingly addresses this problem – we will be able to experiment and tell more about this in the modern and contemporary music context when the code is released.

One of the reasons I chose to use Performance RNN over the other neural network-based polyphonic music generation models is that I find the premise of ready-made performances intriguing. Also, both being projects within Magenta Project, Performance RNN integrates with Onsets and Frames quite well. Performance RNN also preprocesses the MIDI files with extending note-off messages until the end of pedal-off messages from sustain pedal, eliminating the representation of sustain pedal in the input vectors. As mentioned above, Onsets and Frames outputs MIDI transcriptions in a similar fashion – it does not recognize the use of sustain pedal, but it simply extends note durations until they finish resonating.

Another reason is the ease of use; there is step by step explanations of how to train the model on the GitHub page of the project[11]. I followed these steps and ran the commands on Google Colaboratory, which can execute Python[12] code, runs in the cloud and provides free GPU for 12 hours. After 12 hours one can usually rerun the code, though every now and then you are banned from using a GPU for a short period of time, for the purposes of fair share. I connected the Colaboratory environment with Google Drive, so that the model could access the dataset to train on and save

[11] https://github.com/tensorflow/magenta/tree/master/magenta/models/performance_rnn
[12] Python is a high level programming language.

checkpoints during traning. I trained the model for approximately 2 days for each experiment and recorded performance metrics like accuracy and loss with Tensorboard[13].

The model was originally trained with the International Piano-e-Competition[14] dataset in the paper, which consists of the MIDI recordings of the performances of contestants. The dataset includes approximately 1400 pieces of classical music (which I assume to be close to a hundred hours of music for the training set), overwhelming majority being composed by the composers of the common practice period. Before training my model I researched for a rule of thumb for an acceptable amount of data for training, which apparently does not exist. The amount of data varies: Johnson (2015) and Mao et al. (2018) used the Classical Midi Piano Page[15], which is less then 20 hours of data, whereas for the Google Colaboratory version of Music Transformer used over 10000 hours of Youtube videos for training (Simon et al., 2019). However it is doubtful that the outputs of models using more data are musically meaningful. The first output sample given in the blog post for Performance RNN[16] is very impressive in the context of harmonic consistency and phrasing; the fragment shows that the model generalized exceptionally well on the features of consonance, chords, cadences – tonality in general. On the other hand, it seems like an incoherent mixture of musical styles: in thirty seconds a Mozartean texture is overlapped with Chopinesque runs, giving an idea about the most represented composers in the training set. When generating from scratch from the Google Colaboratory version of Music Transformer outputs may even contain circle of fifth clichees in a manner of pop-song arrangements (Figure 2.3). Either way it is not possible to compile sizeable datasets for this thesis because of time and storage constraints but we may benefit from training the model with a smaller

[13] Magenta library is powered by Tensorflow, which is itself an open source library for developing machine learning projects. Tensorboard is a tool of Tensorflow for recording and visualizing performance metrics.
[14] http://www.piano-e-competition.com/
[15] http://www.piano-midi.de/
[16] https://magenta.tensorflow.org/performance-rnn

dataset (as long as we are careful about overfitting[17]) if we manage to get stylistically more consistent outputs as a result.

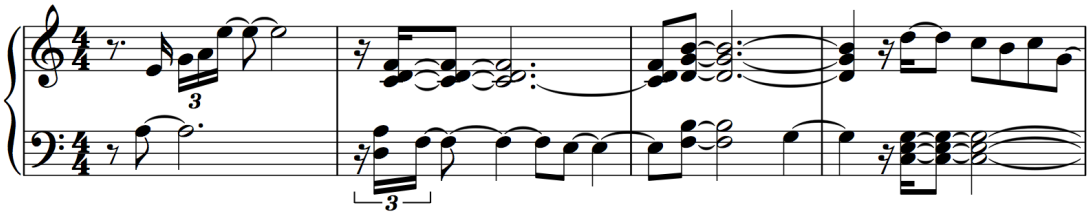


Figure 2.3: Excerpt from a sample output of the Colab notebook of Music Transformer

2.2.1 First experiment

2.2.1.1 Music of Messiaen

Musical output of the 20th century is very diverse in terms of style. Musical modernism is especially marked by its linguistic plurality and the failure of any language to assume a dominant position (Morgan, 1984). Training a network with a dataset including music of various composers of that era is problematic because of the scarcity of common features in the music of different composers, compared to many common features of tonality among the common practice period composers. For that reason I decided to compile a dataset from the piano music of a single composer and I chose Messiaen.

I would prefer to choose a more contemporary composer, however a large repertoire of solo piano music is needed to train the model, which limits the alternatives considerably. Messiaen on the other hand has a significant output of piano music and he has a book called "The Technique of My Musical Language" (1956), in which he summarizes the musical material he employs in his compositions. This book is of great assistance to any musician who intends to analyze Messiaen's music and in this thesis it provides the framework for analyzing the output of the model that is trained by Messiaen's piano music.

In his book, Messiaen discusses his musical material under the subtopics of rhythm, melody, form and harmony. I will take a moment to briefly summarize his

[17] Overfitting occurs when the model does not generalize at all and learns the noise in the training data, in other words, the model memorizes the training data.

explanations, leaving out the discussion about form for the reasons mentioned in the previous section.

First rhythmical tool Messiaen employed is the concept of added value, which are notes of very small durations added to common or previously used rhythmical groupings in a piece. These groupings are often in prime numbers in his music, instead of multiples of 2 or 3 as in the case with the music of common practice period. Messiaen variated rhythms by diminishing, augmenting and retrograding them, which are procedures commonly applied to melodic lines in Baroque music. He was particularly interested in rhythmic groups that remain the same when retrograded, i.e., non-retrogradable rhythms. Other rhythmic interests of Messiaen included polyrhythmic structures, rhythmic canons and pedals.

Melodically, Messiaen emphasised descending melodic intervals of augmented 4th and major 6th, he also used these intervals in cadences. Certain chromatic cells, shown in Figure 2.4, were mentioned in his book. Messiaen is famously interested in bird songs; he used transcriptions of them in his compositions. Other techniques he frequently employed were abrupt register changes, interversion of the order of pitches and the elimination process, akin to the process of fragmentation described in Caplin (1998).

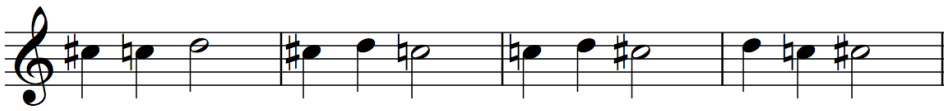


Figure 2.4: Chromatic cells mentioned in Messiaen (1956)

The parallel of the concept of added values in the rhythmical dimension is found as added notes in the harmonic dimension. Messiaen did not put any restrictions on possible additions made to chords, but once more listed the augmented 4th and major 6th as the most frequently added intervals, underlining the importance of them. A more important harmonic concept of Messiaen is his modes of limited transpositions, which are modes that map onto themselves under transposition by certain intervals, so that they have less than 12 transpositions in total. When analyzing the outputs of the model I looked for these modes and chords built from the pitches of the modes, however did not attribute all modes the same level of importance. Messiaen has 7 modes in total – the first two of them are whole tone and diminished scales, respectively. The fifth

mode is contained by the fourth mode so I did not include it in my discussions. I also omitted the seventh mode completely as it is only two pitches short from the chromatic scale. Statistically it covers most musical fragments in the outputs. Modes 3,4 and 6 can be seen in Figures 2.5-7 respectively.

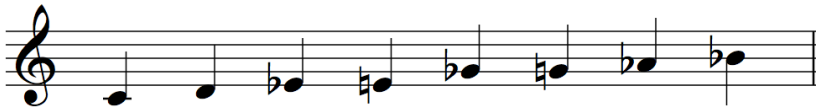


Figure 2.5: Messiaen's 3rd mode of limited transposition

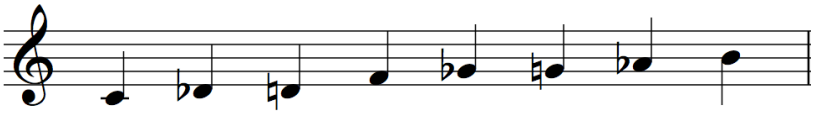


Figure 2.6: Messiaen's 4th mode of limited transposition

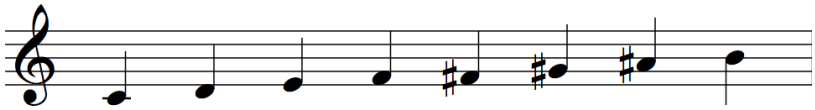


Figure 2.7: Messiaen's 6th mode of limited transposition

Messiaen preferred using these modes over using polytonality, because according to him these modes have fragments from major and minor scales and they give the impression of being in multiple tonalities at once. He freely modulated between these modes and used them in a polymodal context as well.

Aside from these modes, Messiaen also listed particular chords he used frequently. One such chord is the dominant chord, as he called it, which consists of all pitches diatonic to a major scale. Another one is the resonant chord, created by adding the higher partials over a dominant seventh chord. On a related note, Messiaen borrowed the term "effects of resonance" from Paul Dukas, referring to musical passages where chords in higher registers create an artifical resonance for sustained pitches or chords in the lower register. Specific pitch relations for the pitches that create the resonance are not mentioned for this technique.

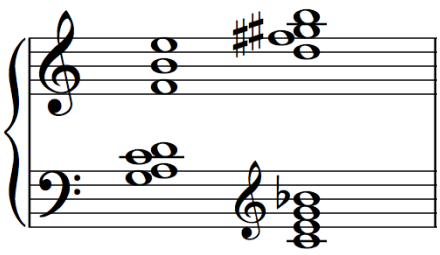


Figure 2.8: Dominant chord and the chord of resonance

2.2.1.2 Analysis of outputs

After the network is trained, any number and length of outputs can be quickly generated. I decided to generate five short outputs of 30 seconds or 1 minutes each for observing the behaviour with different initializations and a longer output of 5 minutes for examining the long-term behaviour in each experiment. I analyzed no further outputs so that I would not end up cherry picking the best musical examples. First two pages of the longer outputs for each experiment can be found in the Appendix. On many occasions the passages in the outputs are not playable by a human performer, but I did not make an analysis from that perspective as the outputs are not directly considered for performance purposes.

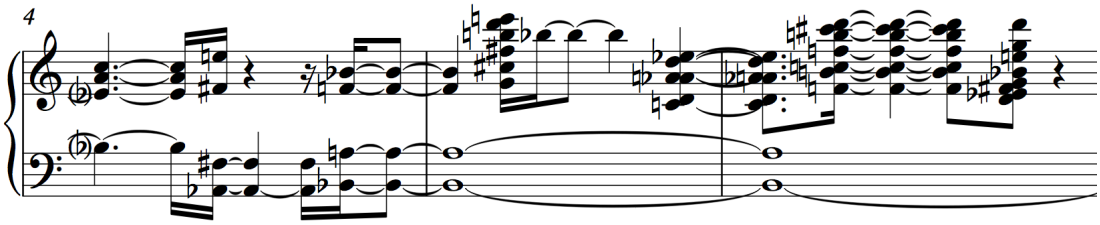


Figure 2.9: Bars 4-6 of the first 30 second-output, first experiment

The first intriguing feature of the first short output of the first experiment is that it starts with a simple sustained major third. Overall, triadic structures commonly found in tonal works are frequently found in the outputs of the first experiment, which is not surprising as the early works of Messiaen (for instance his eight preludes) are mostly composed with triadic material.

Bars 4-6 (Figure 2.9) are interesting for two reasons. Firstly, the melodic interval of augmented fourth, which Messiaen attributes great importance, is found in bar 4 as the top voice moves from E to B flat. This interval is used once more an octave higher in the next bar to establish formal coherence in a very small scale. Furthermore, the root

of the chord underneath is also a B flat, thus it is possible to speak of a cadence in the Messiaenic sense. Secondly, the B flat in the bass and the A on top of it is sustained from the last beat of the 4th bar up until bar 6 and various chords are played in an upper register over these pitches. This fits Messiaen's description (or rather given examples) of effects of resonance.

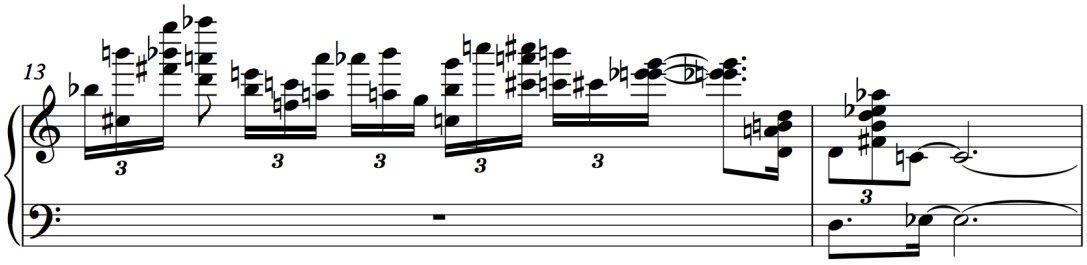


Figure 2.10: Bars 13-14 of the first 30 second-output, first experiment

Bar 13 of the same output (Figure 2.10) demonstrates birdsong like qualities. This impression is mostly due to the high register, consistent rhythm of 16th note triplets, melodic curve and the brief halt of rhythmic activity in the second 8th note of the bar. Frequent register changes also catch the listeners attention. This bar resolves to a diminished scale chord in bar 14, in other words the third transposition of the second mode.

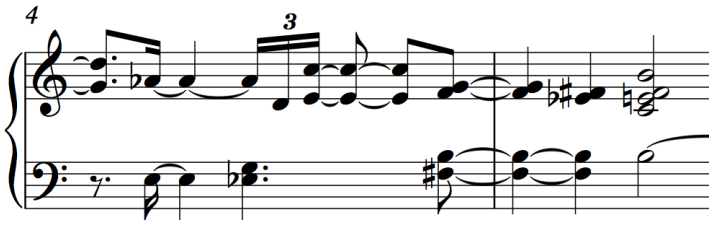


Figure 2.11: Bars 4-5 of the fourth 30 second-output, first experiment

Finding extended passages residing in a single mode is not very easy. Bars 4-5 in the fourth 30 second-output include 8 of the 9 pitches of the first transposition of the third mode and the pitch F, which is out of the mode. My opinion is that the model is not able to internalize a high level feature such as the modes, instead we frequently observe pitch content which almost adds up to a mode, resulting from the interval relationships learned by the model. This is reasonable, as Messiaen does not use the modes consistently and many of the works in the dataset do not use them at all. I believe that with a dataset that consistently uses the modes, the ANN model would be able to learn to compose with them.

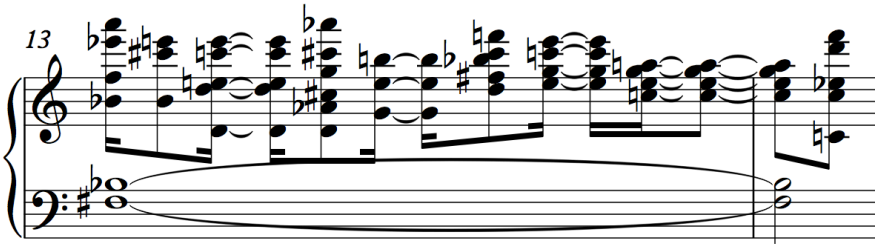


Figure 2.12: Bar 13 of the fourth 30 second-output, first experiment

On the other hand, there are many passages with most of the simultaneities being derived from one of the modes. In the passage in Figure 2.12 the first chord is from the 3rd mode, second chord and third chord are from the diminished scale and whole tone scale, respectively; most of the remaining chords are also derived from these three modes. The last chord of the bar is particularly interesting: it is an altered F sharp dominant seventh chord with an upper structure of a major triad with an added sixth. Also, the whole passage is another instance of effects of resonance.

Aside from techniques and material that belong strictly to Messiaen, there are general musical issues that should be addressed as well, considering the fact that the model is agnostic to any kind of musical idea prior to training. In the fifth 30-second output of the first experiment there are points where musical activity stops, dividing the output into phrases. The model also uses dynamic information for the purposes of phrasing[18]. The beginning of a musical phrase and its velocity information (in a piano roll format) are shown in Figure 2.13 and Figure 2.14, respectively. The velocity of the pitches increase gradually as the phrase begins, creating a crescendo in a musically sensible way.

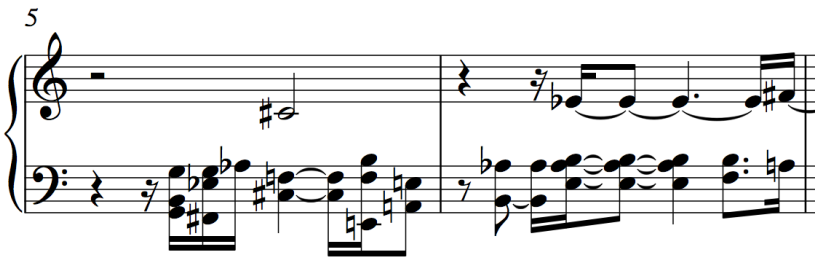


Figure 2.13: Bars 5-6 of the fifth 30 second-output, first experiment

[18] The first experiment is trained with a configuration called “performance with dynamics, compact”. This configuration also uses dynamic information to train the model. The whole range of dynamics is divided into 32 bins, which means that only 32 values are possible for the loudness of a pitch.

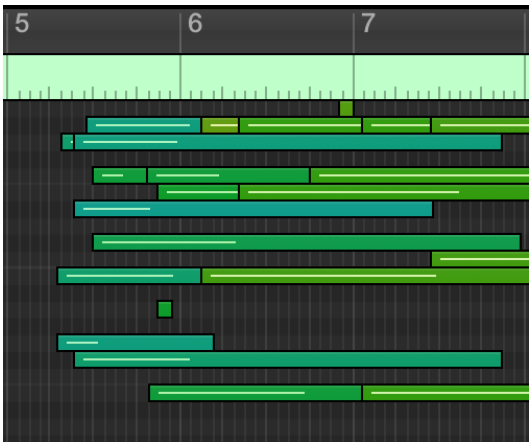


Figure 2.14: Velocity data[19] of figure 2.13

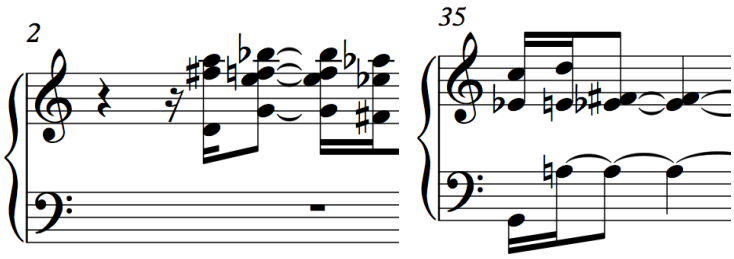


Figure 2.15: Chromatic cells in the 5 minute-output, first experiment

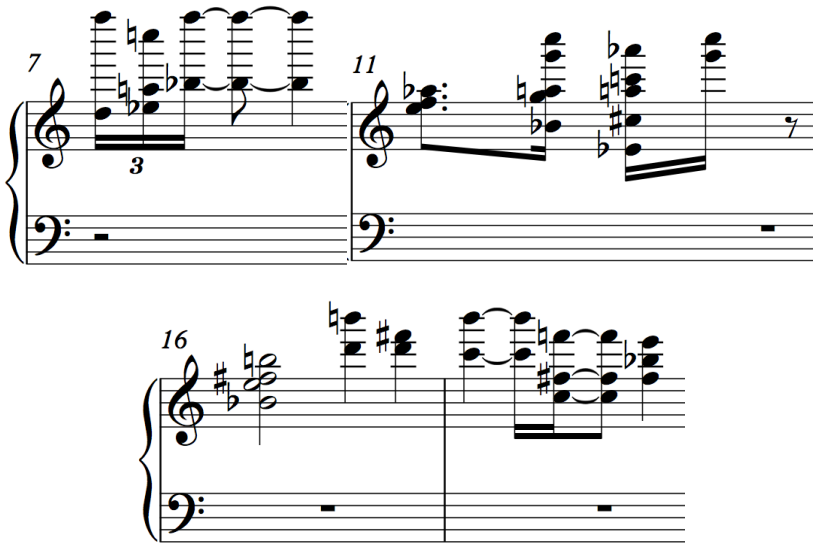


Figure 2.16: Repeated motive in the 5 minute-output, first experiment

The 5 minute-output introduces a couple instances of chromatic cells that Messiaen mentioned among his material. In the 2nd bar the pitches are found in the melody,

[19] The velocity of the pitches increases as the colors in the piano roll get warmer.

whereas in bar 35 another cell is found in the bass voice with the G displaced an octave. A more specific motive that gets repeated three times is seen in Figure 2.16. Within 10 bars we hear the motive consisting of a descending interval followed by an ascending interval of the same size. With each occurence the motive gets rhythmically augmented: from a 16th note triplet to 16th notes and finally to quarter notes. Another repeated pitch pattern is found in bars 44-45 in the bass voice (Figure 2.17). The repetition occurs in a cadence-like moment and the feeling of centricity around the pitch B flat is reinforced with the reiteration of the ascending melodic interval from F sharp to B flat an octave lower in a descending manner.

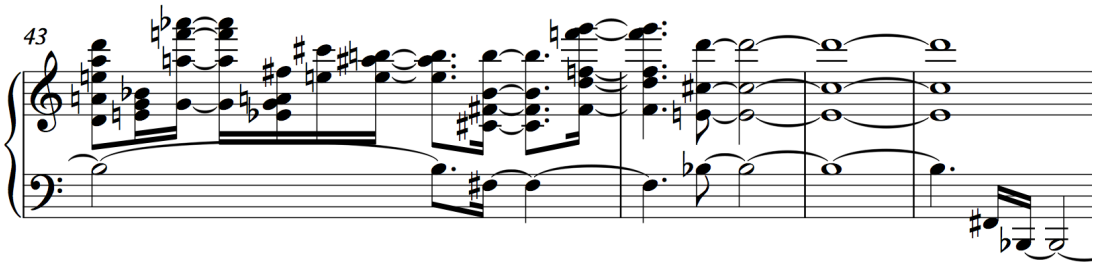


Figure 2.17: Reiteration of the root motion in 5 minute-output, first experiment

As mentioned earlier, fragments of Messiaen's modes fit in major and minor scales and Messiaen uses the modes in a way that the music sounds like in multiple keys at once. This aspect of the technique is examplified in the 5 minute-output in bar 35, where we find a nine-pitched chord involving seven out of the nine pitches of the second transposition of the third mode and two octave doublings. The chord has the pitches F, G, and A flat in it, in other words the first three pitches of an F minor scale. This feature prepares the upcoming F minor material in the next two chords, which can easily be heard and interpreted as inverted versions of F minor extended by 9th, 11th and/or 13th. After the interpolation made by two 4th mode chords, 3rd mode material returns in a different transposition (1st). Then the progression moves to chord with the pitch B in the root. (which is an augmented fourth away from F), namely a B dominant seventh with an added flat 5. Additionally, the smooth voice leading into the initial nine-pitched chord captures the listeners attention as well. Two pitches in the previous chord descends by a half step and the remaining pitch is held (Figure 2.18).

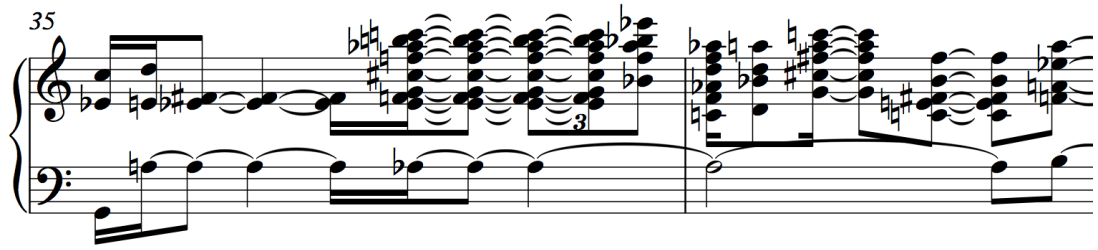


Figure 2.18: Third mode chord and F minor material in the 5 minute-output, first experiment

The last musical example from the first experiment is seen in Figure 2.19, depicting bars 89-90 of the 5 minute-output. The passage shows an extreme registral separation; there are even notes outside the range of piano (marked red) in the lower range. This is possible as the model accepts inputs and outpus for the whole range of the MIDI format, which exceeds the range of piano. Extreme amounts of registral separation is also found in the piano music of Messiaen, examplified by *Catalogue d'Oiseaux, no.7*, bars 19-23 (Figure 2.20). The passage in the original piece is a birdsong transcription of Messiaen. With the part in the higher register being quasi-repetitive in a narrow range, the passage from the output is also reminiscent of a birdsong; however, it is not nearly as consistent in its rhythmic pattern and pitch content. Overall, the model does not seem to capture Messiaen's rhytmhic material like the often used retrograde or non-retrogradable rhythms. The outputs rarely show instances of repetitive rhythmic patterns or any sign of rhythmic development, proving me right about my concerns about the rhythmic framework of the model and the errors made by the software used in the experiment.

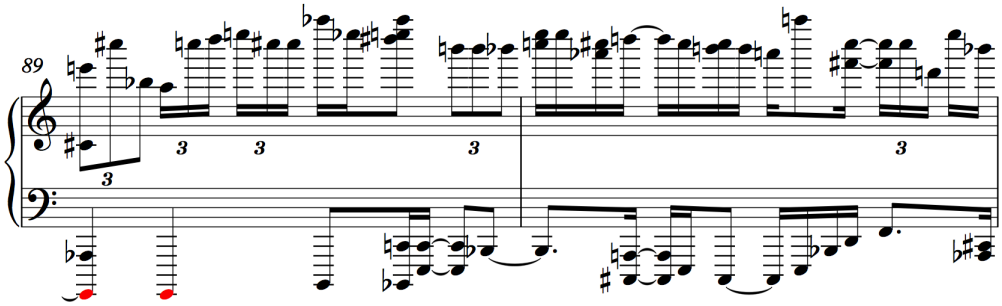


Figure 2.19: Bars 89-90 of the 5 minute-output, first experiment

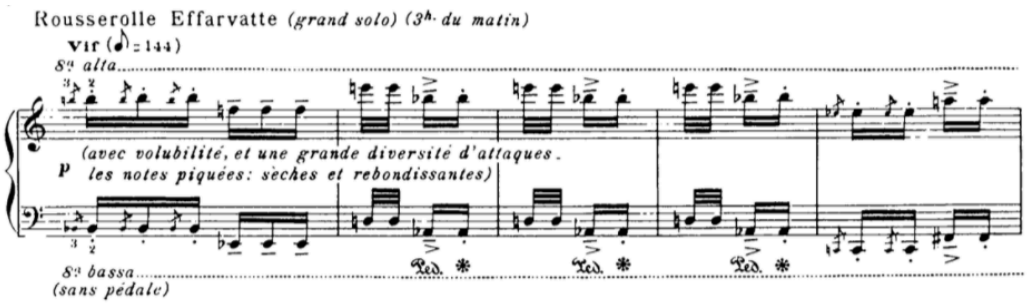


Figure 2.20: Bars 19-23 of Messiaen's *Catalogue d'Oiseaux, no. 7*

Prior to training, I also had concerns about overfitting because of the small dataset I had to use. After assessing the outputs we can assume that overfitting is not an issue, as the generated outputs are very different than the original dataset in various aspects. Figures 2.21 and 2.22 show the training accuracy and loss of the model, with the data collected via Tensorboard. The X-axis depicts the number of steps and the y-axis depicts accuracy and loss, respectively. The outliers and the period around 80k steps with no data is presumably due to reinitializations of the training after a loss of connection to Google Colaboratory.

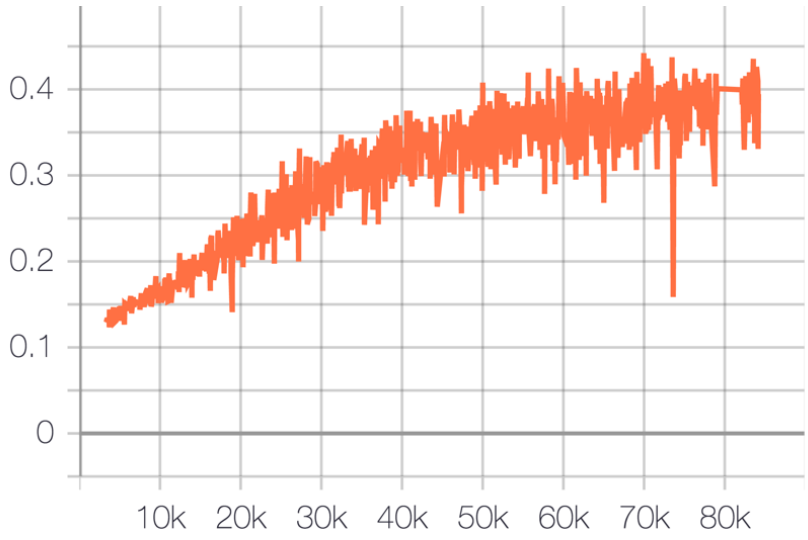


Figure 2.21: Training accuracy of the first experiment

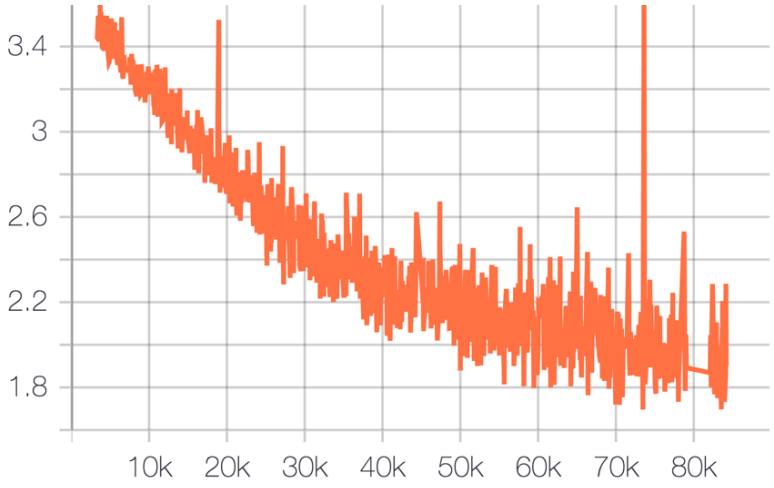


Figure 2.22: Training loss of the first experiment

No data is given about the training accuracy in Oore et al. (2018), however the training loss is smaller than 1 in that experiment. The loss in this experiment is approximately 1,8 at the lowest and the accuracy is as low as 40 percent at best, implying that there is room for further improvement. Aside from the net values of loss and accuracy, we can also make deductions from the constant wide fluctuations of these values. It suggests that the batches are different from each other in various ways and a set of parameters that gets a good result with one batch often does not work as well on the next one. We shall come back to this observation in the next experiment. As increasing the size of the dataset is not possible, the most sensible step is to increase the capacity of the model for the second experiment to get better values of accuracy and loss.

2.2.2 Second experiment

2.2.2.1 Hyperparameters

For the second experiment I decided to change the configuration of the model. I wanted a configuration that does not take dynamics into account, so that the model parameters can focus more on the pitch content. However, only configuration that used compact input files operated with dynamics and the alternatives created very large input files, so I used the same configuration but changed the number of bins for the dynamic information from 32 to 1, indirectly eliminating the dynamics from the model. Furthermore, I added one further layer to the architecture of 512 neurons. Oore et al. (2018) state that the model was not sensitive to the hyperparameter of number of neurons per layer, which is why I did not experiment with that hyperparameter. Lastly,

I set aside ten percent of the original dataset as a validation set. Testing the model parameters on a set of data that has not been trained on enables the user to make deductions about the generalization capabilities of the model and the pitfalls of overfitting and underfitting. The code fragment of Performance RNN for the used configuration is given in Appendix C.

2.2.2.2 Analysis of outputs

The outputs of the second experiment show an increased rhythmic activity in general. Especially the surface rhythm in the lower registers is increased compared to the previous experiment. As a result, many passages seem more chaotic and less musical. My intuition is that the model devoted some of its increased learning capacity to rhythmic precision, which resulted in excessively detailed and unmusical surface rhythms because of the accumulated deviations resulting from the performance, transcription and training phases. Examples of some of the more interesting passages within the outputs are given below. In this experiment the shorter generations are 1 minutes long instead of 30 seconds, except for one output out of the five.

The second of the short outputs examplifies the rhythmic activity in the low registers, with a C/C sharp in the 4th octave being held over for almost eleven measures. Under this held note we can observe triadic structures resulting in F sharp major, F sharp diminished, C major seventh and E flat major chords.

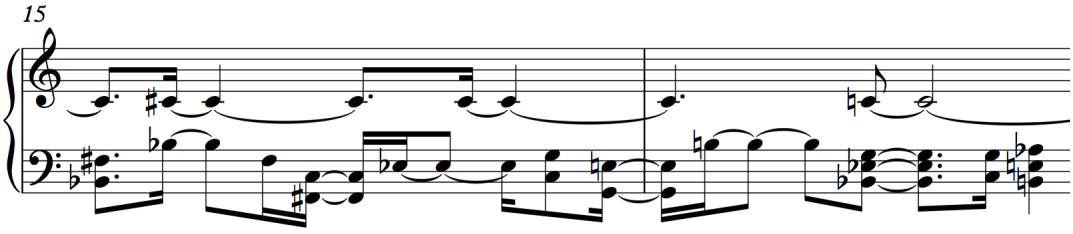


Figure 2.23: Bars 15-16 of the second 1 minute-output, second experiment

In my opinion in many cases the notes which would have been transcribed as a simultaneity in the previous experiment show themselves as broken chords in this experiment. 4th bar of the third short output shows a broken D dominant seventh in this manner, with the frequently observed added note of major 6th on top. Bars 29-30 show a passage with dense rhythmic activity. Two voices almost alternate in their use

of 16th note triplets and regular 16th notes in a contrapuntal texture; however, the result is chaotic as all the examples are transcribed at 120 bpm.

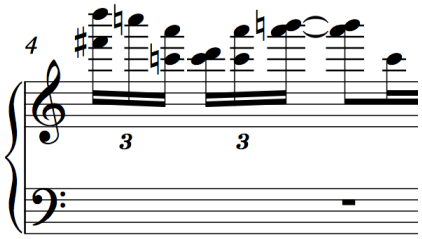


Figure 2.24: Bar 4 of the third 1 minute-output, second experiment

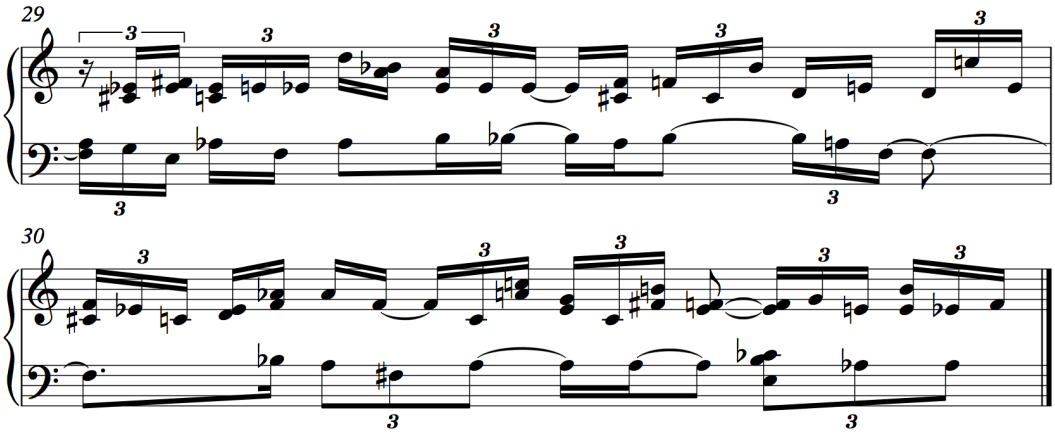


Figure 2.25: Bar 29-30 of the third 1 minute-output, second experiment

The only 30 second-output experiment breathes more than its counterparts. A cadential moment with an upwards leap of an augmented fourth is found in the 2nd bar. The end of the 3rd bar contains a very large melodic leap of almost three octaves. After this single note of high register the melody settles back to mid registers instantly. Messiaen employs large and abrupt leaps in his music as well, so this is not very surprising; however, this leap is not prepared and does not have a musical context, as it is usually the case in these experiments.

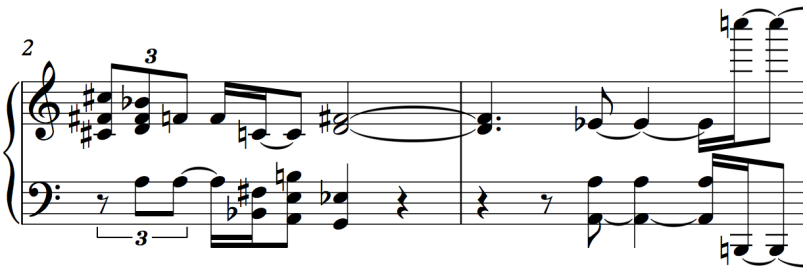


Figure 2.26: Bar 2-3 of the 30 second-output, second experiment

In the same output there is an interesting chord progression in bars 11-12. The progression begins with an F sharp minor chord, with the pedal note A underneath. The second chord is rare in terms of pitch content: its pitches are derived from the fourth transposition of sixth mode. The next two chords point to A minor, followed by an $ii^{\varnothing 7}$ – V – i^{7} progression in E minor, with the V chord replaced by a diminished chord of the seventh degree with an added 6th. The A minor chord and A pedal are understood to be the subdominant chord and pedal in retrospect. This progression is underlined by the root motion from A to E in the very next bar.

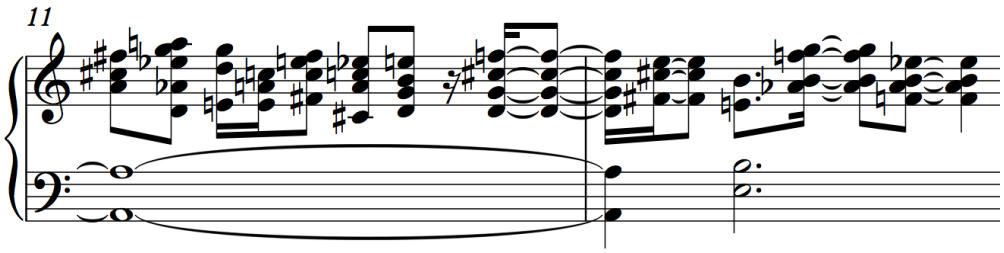


Figure 2.27: Bar 11-12 of the 30 second-output, second experiment

Moving on to the 5 minute-output of the experiment, bars 35-36 is the first passage I find worthy of commentary. The displayed passage (Figure 2.28) begins with a shift in register in the third beat, notably from a chord built with the intervals of augmented fourth and major sixth on top of the pitch A. Then, we observe numerous repetitions of pitches C and C sharp in a short space. This is just one example of many instances where pitches within a narrow range are repeated rapidly, creating a frivolous rhythmic activity.

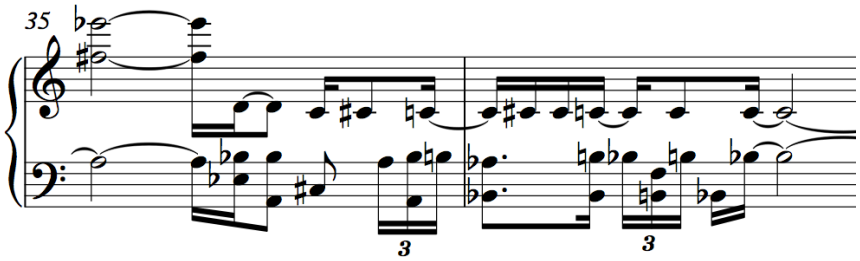


Figure 2.28: Bars 35-36 of the 5 minute-output, second experiment

Another recurring concept in the outputs is the interpolation of the harmonic functions. In bar 58, chords that can be mapped to either of mode three or six are interpolated by an F minor hybrid chord. In bar 94, the C minor chord with an added eleventh comes

back two bars later in the last beat. In between there is a diminished scale chord and an F sharp minor chord, the latter one being introduced with a wide leap.

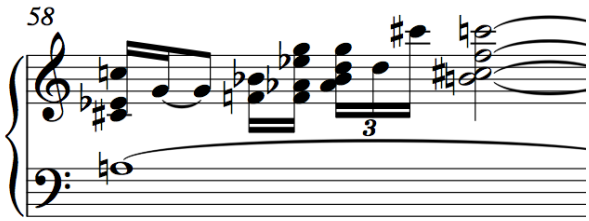


Figure 2.29: Bar 58 of the 5 minute-output, second experiment

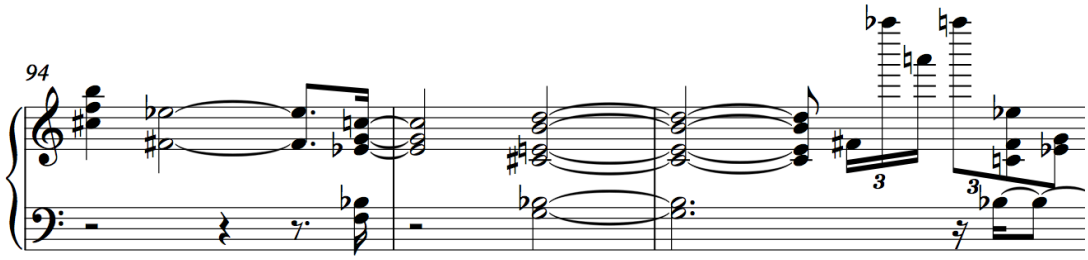


Figure 2.30: Bars 94-96 of the 5 minute-output, second experiment

Two of the special chords appear in the long output of this experiment. In bar 64, we can see the chord of resonance with the 5th and 11th of the chord omitted. However, the chord is voiced in a way that the chord notes form two distinct intervals of augmented fourths. In bar 138 the chord in fourths can be spotted, which is a chord built of alternating perfect and augmented fourths, accumulating to the pitch content of the 5th mode. In this case the chord is two notes short of the fifth mode, but it is voiced exactly as Messiaen describes. The chord in fourths has a B flat in as its root and it is preceded by a B flat dominant seventh chord with the added notes of 9th, sharp ninth and sharp 11th. It is followed by G minor (with an added 9th) and B flat minor seventh chords. Overall, we can claim that triadic material is frequent in the second experiment as well.

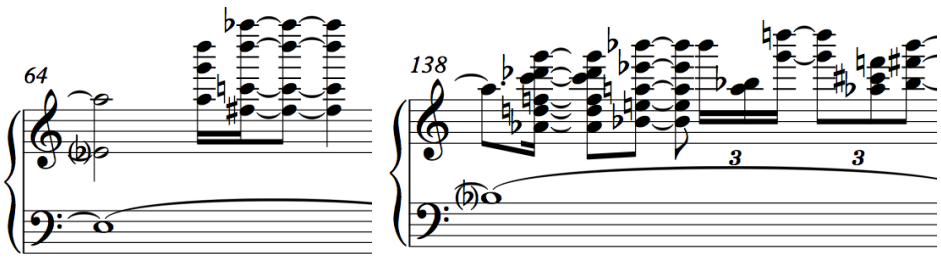


Figure 2.31: Special chords in bars 64 and 138 of the 5 minute-output, second experiment

Last musical example given from this experiment is the most naively constructed passage in the outputs of both experiments. It is a lullabyish melodic idea centered around the pitch E flat. This is the first fragment in which we observe a very consistent repetition of a melody with clear direction. It can also be argued that the duration of the pitches are extended with added values to create a slight rhythmic imbalance.

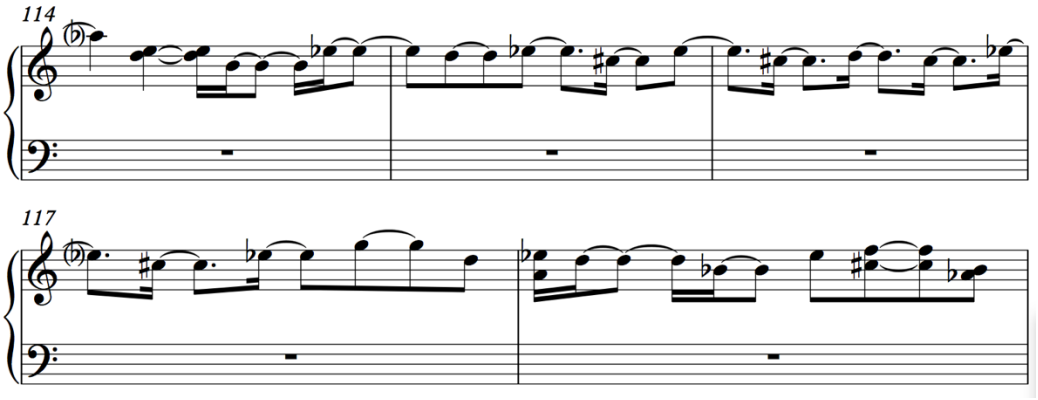


Figure 2.32: Lullabyish musical idea in bars 114-118 of the 5 minute-output, second experiment

Figures 2.33 and 2.34 show the accuracy and loss graphs of the training and evaluation sets in orange and blue, respectively. The straight lines in the blue curve denote the time intervals where no data is gathered[20]. The intuition in the last experiment was that the model was not overfitting. In this experiment, we see an increase in the accuracy of the training set. However, the data show that the accuracy/loss of the validation set decreases/increases very early (after approximately 2-3 thousand steps of training). This proves that the model is not able to generalize and it is in fact unsuccessfully "trying" to overfit. It can be claimed that a more refined dataset is needed for better results. We had a hint in the previous example from the wide fluctuations of accuracy and loss at every step. Improvement in the transcription data could help, but if we desire from model to find common features in the data, we primarily need to use a dataset that employs the discussed features more consistently. Limiting the used repertoire of Messiaen within a time period could be a solution, but in that case the size of the dataset shrinks considerably. The ways of compiling a more consistent

[20] I ran an evaluation job in the Google Colaboratory simultaneous with the training job. However, once in a while, the evaluation job throws an error and stops without an apparent reason. I was not able to see a reason in the code of Magenta and did not spend more time on the issue as I am not a programmer and there was enough data to see the general trend.

dataset for 20th or 21st century repertoire is a possible topic of discussion in further studies.

Having mentioned this issue, I think the model still generates original and musically interesting results as it is not able to overfit to the dataset properly. The accuracy for the validation set is not released in the paper of Performance RNN as well - the measure may not be deemed relevant to the authenticity of the generated outputs. For the purposes of this thesis, I moved forward by repeating the experiment with same hyperparameters, but I used an expanded dataset including works from composers other than Messiaen.

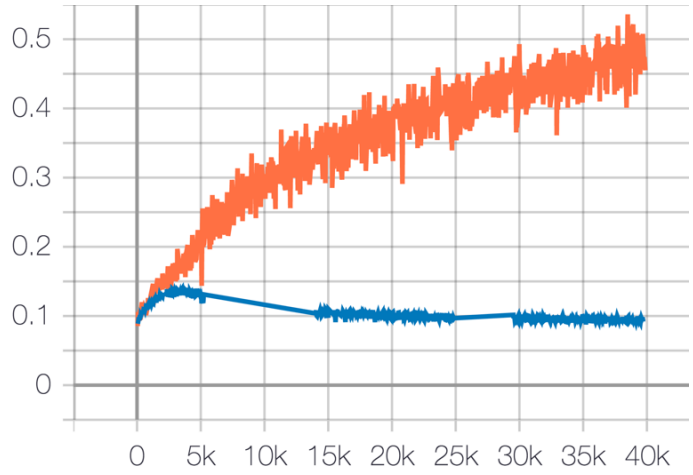


Figure 2.33: Training accuracy of the training (orange) and validation (blue) set, second experiment[21]

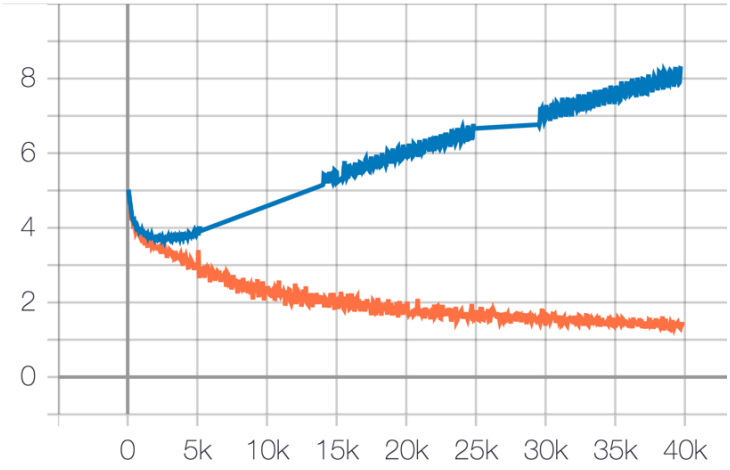


Figure 2.34: Training loss of the training (orange) and validation (blue) set, second experiment

[21] The model was trained further, however the accuracy/loss data was not collected as the trends of the curves were already clear.

2.2.3 Third experiment

2.2.3.1 Music SOM

As discussed earlier, expanding the dataset in the previous experiments with works sharing similar features is problematic for the modernist repertoire and for contemporary music, if not impossible. The decision of which composers to add may be done completely by subjective judgement; however, I decided to use a more theoretical approach and used a neural network model called Self Organizing Maps.

Self Organizing Maps (SOM) are unsupervised learning models proposed by Kohonen in 1982. SOM is a tool for visualizing high dimensional data. This visualization also displays the similarities among the input data (Kohonen et al., 2012).

Before moving further, I want to clarify the term "unsupervised learning". In our previous models we used supervised learning techniques: in every training step there was a correct output vector for every input vector. We measured the distance between the actual input of the model and the correct output and gave our model a feedback accordingly; so that the weights are updated in the right direction. To be clear, the correct output vector is the next input vector in the sequence, as we are training the model to recreate the musical example we feed to it. In the case of SOM however, there is no measure of error that we can backpropagate to update the weights.

The architecture of SOMs consists of a single input and output layer. The neurons in the output layer have weights of the same dimension as the input vectors. The learning takes place in three steps after these weights are initialized. First step is called competition. The distance between the weights of each neuron and the input vector is measured with the help of a distance function – the neuron with the least distance wins. The second step is cooperation. In this step we choose the neighbours of the winning neurons, primarily according to their distance in the two dimensional output space. The weights of the winning neuron and its neighbours are updated in the third step, which is named adaptation (Khazri, 2019).

I used a code called Music-SOM submitted to the software development platform GitHub in 2017 by Odysseas Krystalakos. The raw audio dataset of the piano compositions that I compiled included works by Messiaen, Karlheinz Stockhausen, Tristan Murail, Luciano Berio, Second Viennese School (mostly Schönberg with an addition of a couple works by Berg and Webern), Alfred Schnittke, Brian

Ferneyhough, Pierre Boulez and Salvatore Sciarrino. Each composer had a hundred samples of 30 seconds each. For this experiment I chose composers with a significant amount of solo piano compositions who work outside of tonal idiom. Three of these composers (Stockhausen, Boulez and Murail) have had Messiaen as their teacher at some point, though their styles did not always resemble each other. My plan was to choose four of the eight composers mentioned above and to increase the database to more than 10 hours by adding an hour of piano music from each new composer.

The input vectors for the Music SOM are 35 dimensional. The features are extracted from the audio files with the help of a Python package called LibROSA, which is used for music and audio analysis ("LibROSA", n.d.). The first dimension is reserved for the beat tracking function. The function returns the estimated tempo and beat locations. The second dimension is for the tuning estimation function, which returns the estimation of tuning deviation from the reference of A=440 Hz (McFee et al., 2015). Next seven numbers are the tonal centroids as calculated by Harte et al. (2006), which is a representation of pitch and harmony based on Riemann's Tonnetz. Tonal space is six dimensional according to Harte et al.[22], thus I am assuming that the seventh dimension represents the time frame that the data is gathered in. Mel Frequency Ceptral Coefficients (MFCC) cover twenty one slots of the input vector. These coefficients are used to describe the overall shape of a spectral envelope and therefore convey important information about the timbre (Tjoa, 2015). Remaning slots are filled by the feature of spectral contrast, which considers the spectral peak, spectral valley and their difference in every sub-band (Jiang et al., 2012).

I have made adjustments on the code of Krystalakos. Aside from minor updates[23], parameter changes (concerning the map size and audio samples per composer) and required fixes for plotting a larger map than 4x4 (which was the size of the original example given by Krystalakos), I placed the calls for the training function in a loop. In the original version of the code the training function feeds every sample into the model only once, whereas my version shuffles the training samples and feeds them to the model 250 times, aiming for better results with a longer training period. Both versions can be found in Appendix E. Note that the function is called multiple times

[22] Two dimensions each for perfect fifths, major thirds and minor thirds.
[23] Compiling the Python code in a newer version than the code was written in often requires some updates to various commands.

even if every sample is fed into the model once. This way the learning rate (the third argument passed to the training function) can be reduced as the training progresses.

I trained the model with a grid of 20x20 neurons. To decide which composers to choose, I examined the neurons which are won by the audio files of Messiaen works more than 200 times[24]. A sensible measure of similarity is the number of wins of the other sample groups in these neurons, as the wins are decided by the similarity of neurons and samples in the first place. The generated maps will be different in each run as the weight initializations and sample order shufflings are random, which is why I repeated this experiment twice and compared the results. Table 2.1 displays the total number of wins of each sample group in the neurons won by Messiaen audio files by more than 200 times for both experiments and Figure 2.35 shows the map resulting from the first experiment.

	KS	**LB**	**SV**	**SS**	**PB**	**BF**	**AS**	**TM**
1	5975	5054	6403	5000	4554	5925	5703	5708
2	6159	7243	6539	6014	4743	6339	4762	5724

Table 2.1: The number of wins in cells won by Messiaen more than 200 times[25]

The choice of composers became easier with the help of the table in figure 2.6, as Stockhausen, Ferneyhough and composers of the Second Viennese School are amongst the top four in both experiments. As the fourth composer to add to the dataset, I chose Tristan Murail based on subjective judgement. He is one of the pupils of Messiaen and his compositions frequently use resonance effects in Messianic sense, high-pitched material and register separation. Moreover, considering Messiaen's emphasis on resonance and his use of different registers, we can claim that he, like Murail, has a spectral sensitivity in his music.

[24] The data of the amount of times each neuron is won by each sample group (composer) is available.
[25] In figure 2.6 and 2.7 the composers mentioned earlier in this section are referred by their initials. SV denotes Second Viennese School.

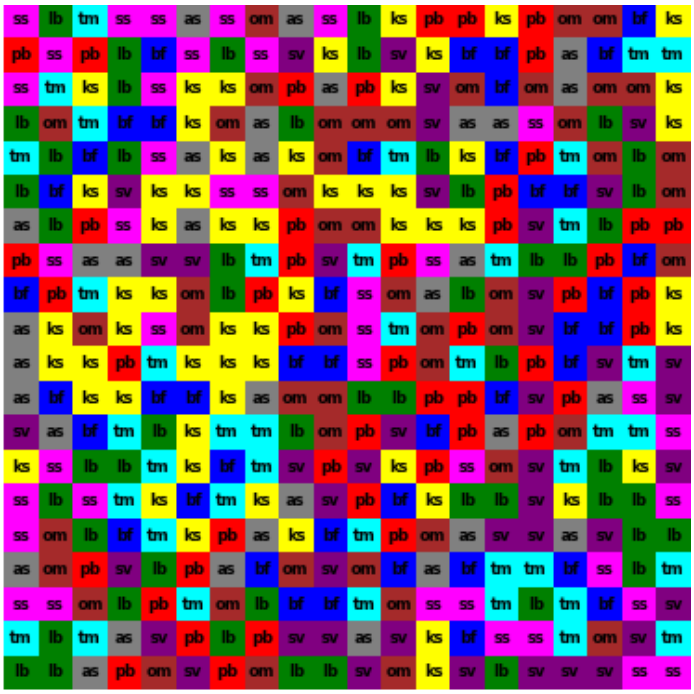


Figure 2.35: Map from the first experiment with Music SOM

The experiment is carried with the same hyperparameters from the second experiment. If I had been able to analyze the outputs of the second experiment before I started training for the third experiment I would have considered reversing the changes done in the second experiment; however, that was not possible because of time constraints.

2.2.3.2 Analysis of outputs

After training the model with a new dataset including the works from multiple composers, the natural expectation is observing musical features from the works of those composers in the generated outputs. This is what I looked for when analysing the outputs of this experiment.

The first musical feature unobserved in the previous outputs is shown in Figure 2.36. In the first two experiments, end of phrases or distinct groups of notes were marked by sustained pitches of long duration. In this example, the last note is a B flat in the 3rd octave, which functions like a punctuation mark abruptly ending the phrase. It is actually even shorter than a 16th note[26], but more importantly it is in a different

[26] It is possible to quantize the MIDI file to 32nd notes or lower subdivisions, but the scores become very hard to read for the purposes of analysis.

register than the previous notes, which makes this punctuation effect even clearer. Similar passages with sudden changes in note duration within the notes which are musically grouped together are frequently found in the outputs of this experiment. We can attribute this novelty to the inclusion of Stockhausen, Schönberg and Ferneyhough to the dataset.

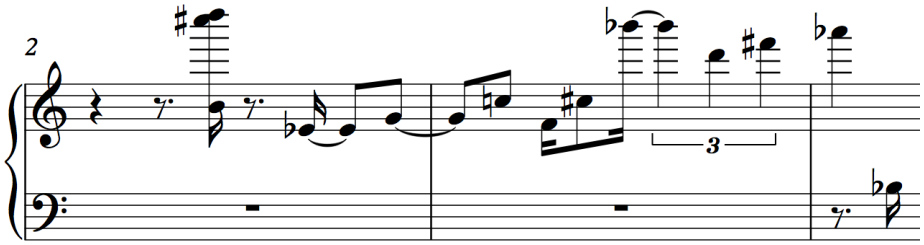


Figure 2.36: Bars 2-4 of the first 30 second-output, third experiment

In the 13th and 14th bar of the second 30 seconds-output (Figure 2.37) there are two consistently ascending adjacent arpeggios, which were also a rarity in the previous experiments. The second arpeggio has a rubato-like rhythm in the manner of a brief rhythmic decrescendo and crescendo (not noticable in the transcription), which is reminiscent of the music of Tristan Murail, for instance of *Les Travaux et les Jours*. An examplary passage from that piece is given in Figure 2.38.

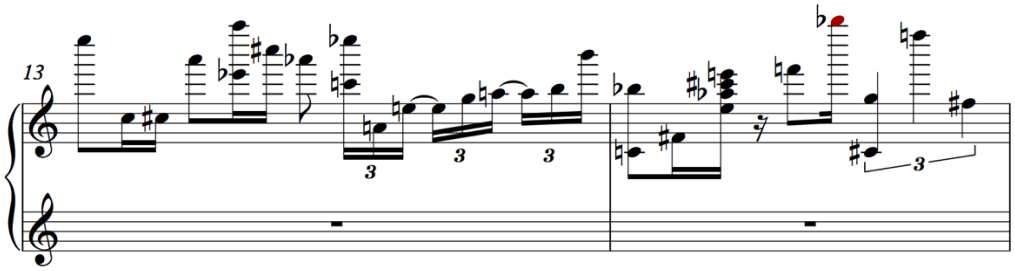


Figure 2.37: Bars 13-14 of the second 30 seconds-output, third experiment

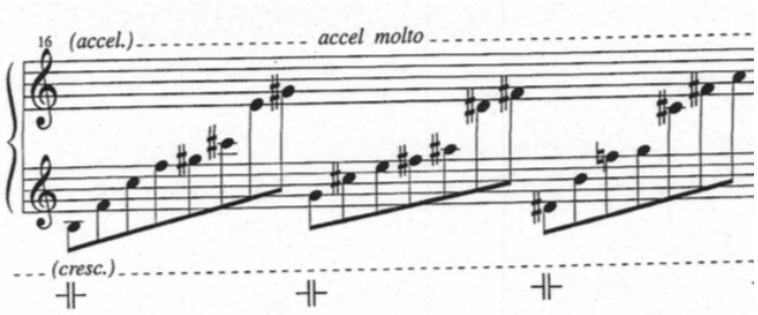


Figure 2.38: Bar 16 of *Les Travaux et les Jours, no.1*

The third short output of the third experiment is perhaps the most interesting one among all outputs of this experiment. The pitches D/D sharp are rapidly played in a manner of tremolo (occasionally speeding up and down in the process, which could be more successfully notated in rhythmic crescendos and decrescendos) for four bars before unveiling a texture of arpeggios (Figure 2.39), once more reminiscent of Murail. Looking at this passage, we can claim that the model has reacted to the additions made to the dataset. Works of Messiaen do not have passages of incessant repetition, unlike *Klavierstück IX* of Stockhausen (Figure 2.40) and *Territoires de l'oubli* of Murail, both among the additions made to the dataset for the third experiment.

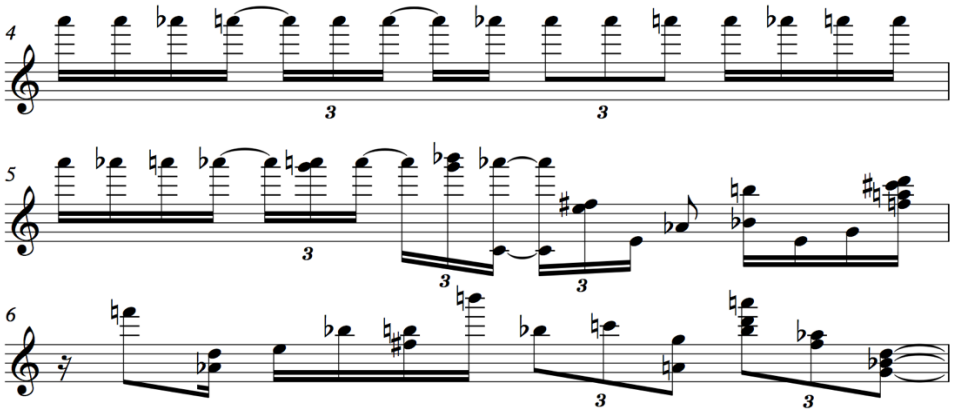


Figure 2.39: Bars 4-6 of the third 30 second-output, third experiment

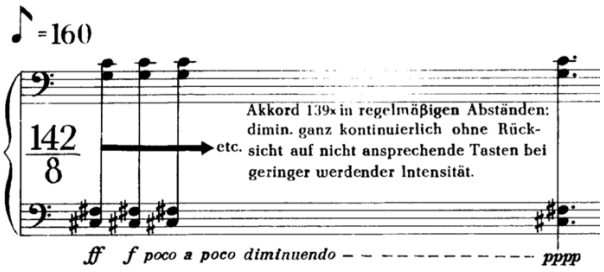


Figure 2.40: First bar of *Klaviersütck IX*

In previous experiments there were many examples of abrupt register changes. The long output of this experiment contains extreme examples of register changes in which a single note is played per each register (Figure 2.41), reminiscent of the pointillistic style of Webern. Overall, chordal textures still dominate, but compared to previous experiments there are more instances where it is closer to a two-voice counterpoint (Figure 2.42), in the style of Stockhausen's *Klavierstück IV*.

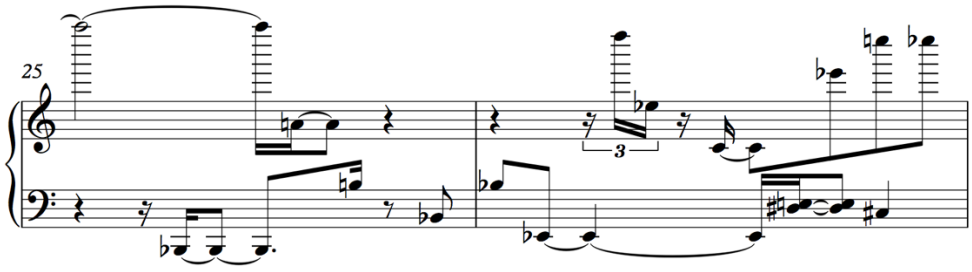


Figure 2.41: Bar 25-26 of the 5 minute-output, third experiment

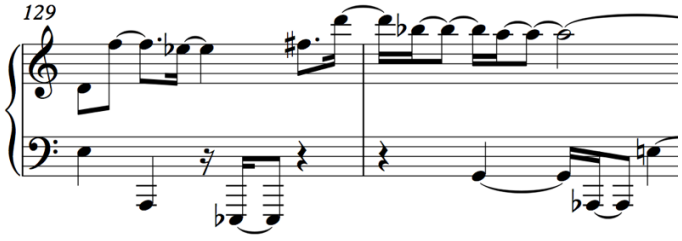


Figure 2.42: Bars 129-130 of the 5 minute-output, third experiment

Considering the fact that serialist composers are added to the dataset for this experiment, adopting a set theory apporach for the analysis makes sense as well. For instance, following the terminology of Straus (2005), the pitch class set found in bar 23-24[27] of the output with the prime form of (012579) is transpositionally equivalent to the one found in bar 41. I also looked for instances of chromatic saturation, though I did not find any. Instead, there are many passages with different pitch classes accumulating quickly to undermine the sense of centricity, like when 10 different pitch classes unfold in a couple beats in bar 141. Deeper analysis could reveal more intricate relationships, but analysing in the same level of depth as Toop (1990) did for Ferneyhough's *Lemma-Icon-Epigram* would be speculatory, as it does not seem realistic for the model to learn concealed features such as filtering (Toop, 1990, pg.59) from a dataset with only an hour of Ferneyhough's music out of a total of ten hours.

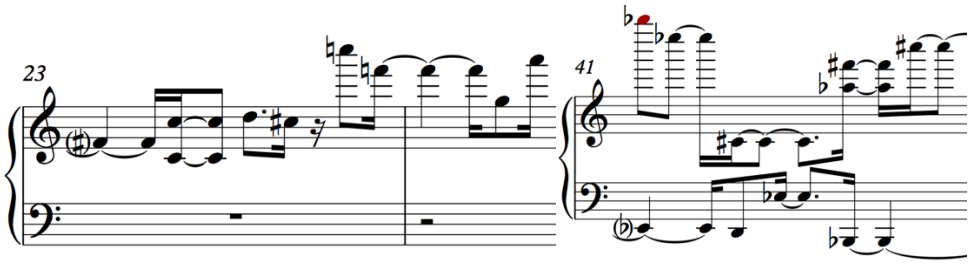


Figure 2.43: Transpositionally equivalent pitch class sets in the 5 minute-output, third experiment

[27] Consisting of C, C sharp, D, F, G and A.

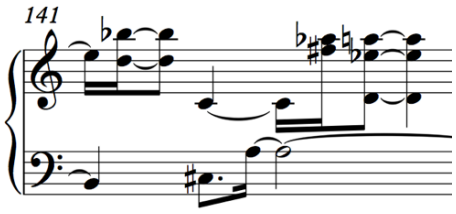


Figure 2.44: Bar 141 of the 5 minute-output, third experiment

In general, we can say that the experiment is successful, as we are able to observe different musical features in the output that we can relate to the works of the composers added to the dataset. It would be particularly interesting to train the same dataset with a configuration that handles dynamic information as well, considering the dynamic changes of Stockhausen within the idiom of total serialism or the level of detail in Ferneyhough's music.

Looking at the graphs of accuracy and loss, we can state that there are no major changes in the behaviour between the second and third experiment, except for the fact that the accuracy graph is more unstable in terms of jumps between batches. This is expected considering that the dataset in this experiment is more diverse. More importantly, a small increase/decrease can be observed in the level of accuracy/loss of the validation set. The peak/trough levels of both curves are also seen a bit later in the training compared to the second experiment. From this information we can retrospectively understand that the size of the initial dataset was indeed insufficient. Even the extension of the dataset by the works of other composers provided an increase in the generalization capabilities of the model.

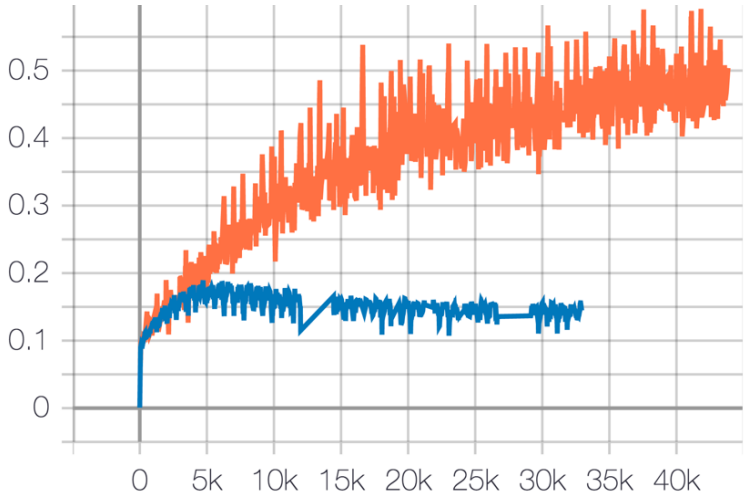


Figure 2.45: Training accuracy of the training (orange) and validation (blue) set, third experiment

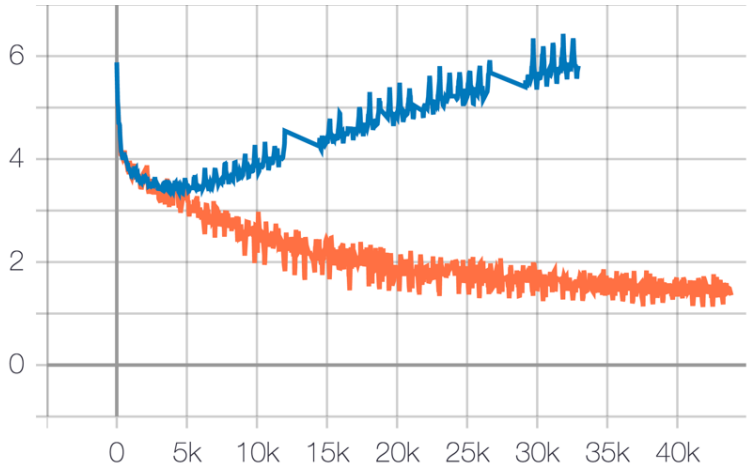


Figure 2.46: Training loss of the training (orange) and validation (blue) set, third experiment

2.2.4 Fourth experiment

2.2.4.1 Improvisational context

For the last experiment I decided the train the model with my own improvisations and use the outputs to compose a piano miniature. I improvised for approximately six and a half hours in 28 sessions, which is close to the length of Messiaen's piano repertoire. The length of the sessions change between 5 and 25 minutes. To be consistent with the previous experiments I recorded audio files and converted them to MIDI with Onset and Frames afterwards, instead of recording MIDI files directly. A sample of these improvisation transcriptions can be found in Appendix G. Finally, I used the model parameters from the first experiment as I thought dynamic information in the outputs could be helpful for the subsequent composition process.

The improvisations were not totally free, in the sense that there was musical material prepared beforehand. This material can be grouped in categories of vertical and horizontal. The first item in the vertical category is polychords. Frequently used polychords include major triads a minor 2nd, major 2nd, minor 3rd, major 3rd or tritone above another major triad, and minor triads a minor 2nd, minor 3rd or tritone above a major triad. The second item is the natural or scaled versions of the harmonic spectrum, with the scale factors 0.8, 0.9 and 1.2[28]. The scaled spectrum examples over the fundamental C2 are found in Figure 2.47. Root motion by major and minor third is favored when progressing between these material. The remaining vertical sonorities

[28] The scaled versions of spectra is adjusted to equal temperament.

include extended chords found commonly in jazz music and other sonorities derived from material in horizontal category. The items in the horizontal category are 1st, 2nd (diminished and whole tone scales), 3rd and 6th modes of Messiaen, which are presented earlier in this thesis.

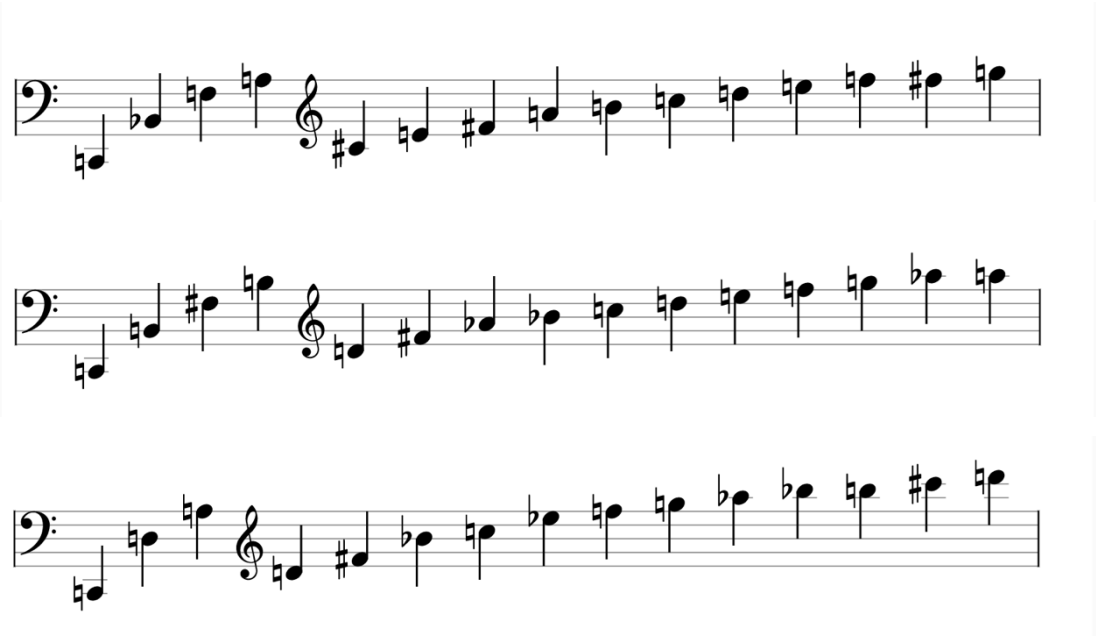


Figure 2.47: Spectra stretched with scale factors 0.8, 0.9 and 1.2, respectively

The resulting improvisations however do not consist entirely of the material discussed above. Deviations occur, because I could not afford to be selective while compiling this dataset, though I was not always satisfied with the recordings in terms of aesthetics.

2.2.4.2 Compositional thinking

Before moving on to the compositional process, I would like to briefly talk about the outputs of the model. Compared to the outputs in the previous experiments, the outputs in this experiment more frequently have simplistic passages containing triadic material (Figure 2.48). Overall, the outputs seem harmonically and rhythmically less complex, demonstrating a more improvisatory character involving many arpeggios and runs, not unlike improvisation in the context of jazz music (Figure 2.49). These features of the outputs are reflective of my capabilities as an improvisor and experience in jazz improvisation.

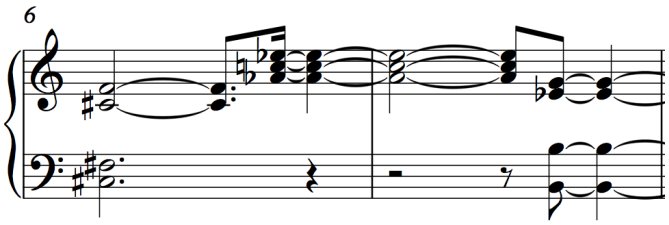


Figure 2.48: Sample output containing triadic material

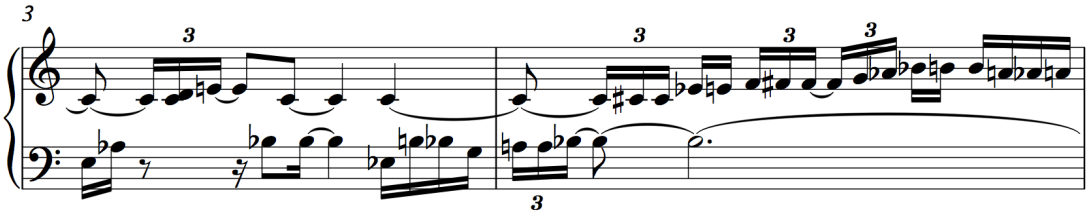


Figure 2.49: Sample output containing a single-voice run

Looking at the training accuracy, we observe a similar trend to the previous experiments. The main difference is that the accuracy curve is much more stable, which indicates that the improvisation dataset was more coherent in terms of style. This is expected, as I worked with limited material while improvising. Also noteworthy is the level of training accuracy, which is slightly lower compared to the third experiment.

Taking these graphs into account, further analysis of the outputs is avoided in this experiment – the capabilities of the model have already been demonstrated in the previous sections. Instead, the focus is shifted to the compositional process.

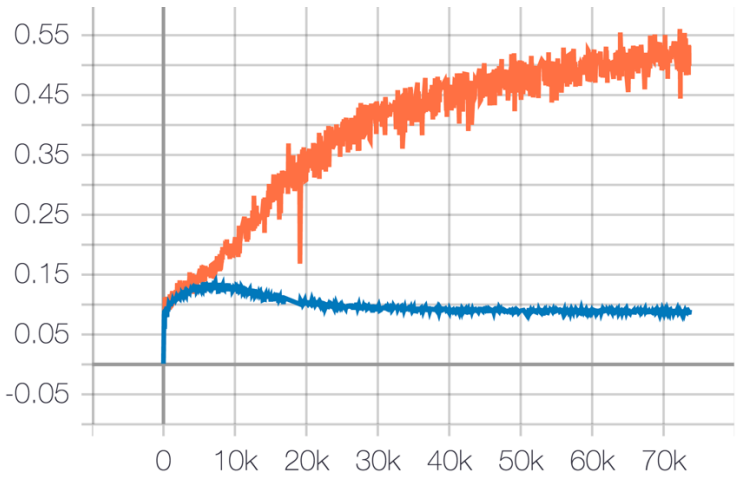


Figure 2.50: Training accuracy of the training (orange) and validation (blue) set, fourth experiment

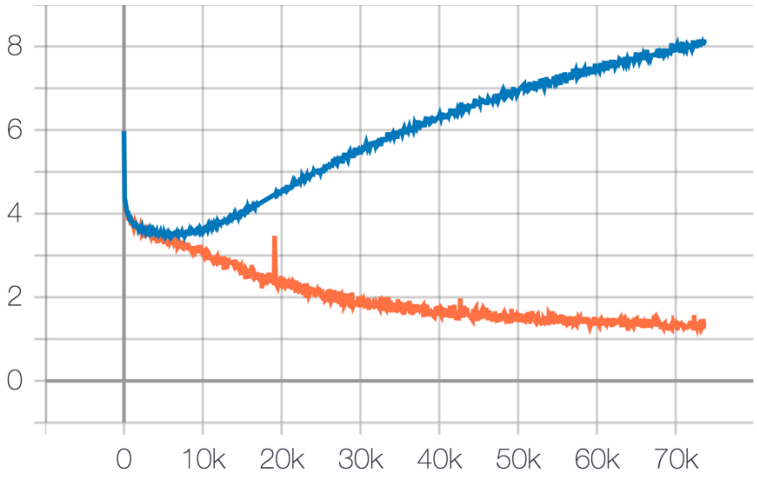


Figure 2.51: Training loss of the training (orange) and validation (blue) set, fourth experiment

As I mentioned earlier, outputs can be generated in a couple of minutes after the training is complete. In a compositional context, this means almost unlimited material to work with. Because of time constraints, I chose to generate fifteen one minute long and one five minutes long output, totaling to twenty minutes of material. Then I went on to collect fragments out of these outputs which particularly strike me as aesthetically pleasing or formally coherent. These fragments range between 2 to 10 bars and are over four minutes long in total.

At this stage I would like to point out that the ways of composing with this material is practically unlimited, it depends on the choices and working habits of the composer. For a longer piece and a larger time window for composing I would work with shorter material and develop more. For the purposes of this thesis however, I chose to stay closer to the material in the outputs to demonstrate the usefulness of them. Also, this choice helped me to compose more quickly as I had to make fewer lower level decisions about pitch content and rhythm, and intrigued me aesthetically as the end product preserved the improvisational character in the outputs.

The end product, which I called “comprovisational miniature for piano”, is approximately two minutes and fifteen seconds long. The score can be found in Appendix I. Specifically, the piece is composed juxtaposing the fragments I extracted from the outputs. Among these fragments, I selected passages which were suitable for creating a formal coherence and made changes to create smooth transitions whenever necessary. Dynamics, articulations and pedalings are inserted throughout the

composition and adjustments are made in the notation for the sake of readability and playability. I would like to look closely at the juxtapositions and changes (made to the outputs) in the composition before I conclude the section.

The first five bars of the piece is extracted from the first bars of fourteenth 1 minute-output, with the difference of a single pitch in bar 5. The original pitch of F sharp is raised a semi-tone to G to obtain the dissonant interval of a tritone instead of a perfect fourth. Also, one beat is added to the bar to space the material out. Extensions of durations of chords or single notes and insertion of rests is employed frequently throughout the piece to let the material breathe. The next fragment from bar 6 to 14 is found in the seventh 1 minute-output and serves the purpose of leading the low register way up to the fifth octave of the piano. The last pitch of A5 found in bar 14 is originally a Bb4 in the output, but it is changed for voice leading and register shifting purposes. The next fragment in bars 15-16 is reused in bars 24-25. It is subjected to various alterations in pitch and rhythm; the original version can be seen in Example 2.52. The last chord of the passage is held over the next fragment for a smooth transition in the composition.

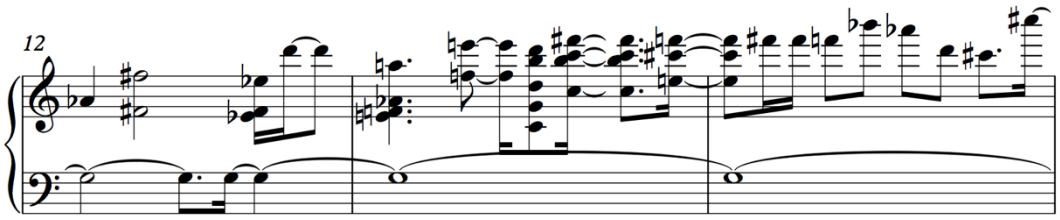


Figure 2.52: Bars 12-14 in the fifteenth 1 minute-output, fourth experiment

The next fragment between bars 17-23 (original output in Figure 2.53) is taken from the earlier measures of the same output. The noteworthy changes in the fragment are the pitch F6 found in bars 17 and 18, which is inserted for register considerations and the quintuplet found in bar 20. In faster passages or passages with unstable rhythm, I occasionally felt the need to stabilize the rhythm with consistent durations. Similarly a septuplet is used in the next fragment from the thirteenth 1 minute-output (bars 26-31, original passage in Figure 2.54). The rhythmic structure in bar 30 is altered with the inserted 16th note triplet and sextuplets. This change contributes to building a momentum at this point, as it provides an increased surface rhythm.

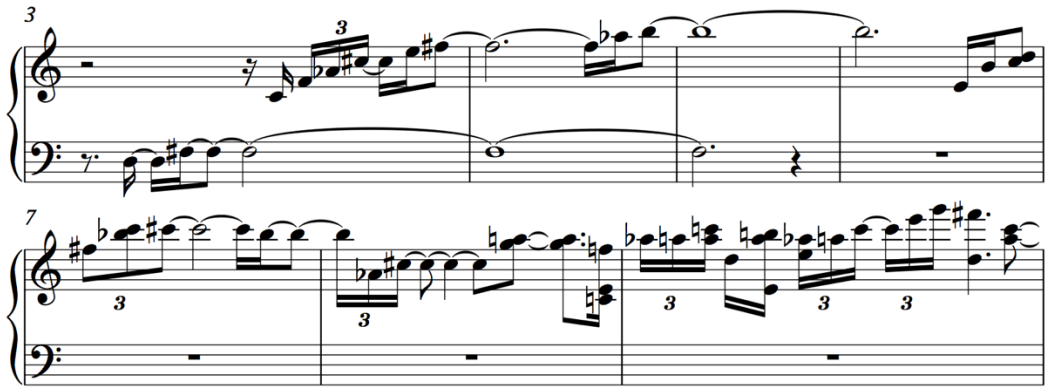


Figure 2.53: Bars 3-9 in the fifteenth 1 minute-output, fourth experiment

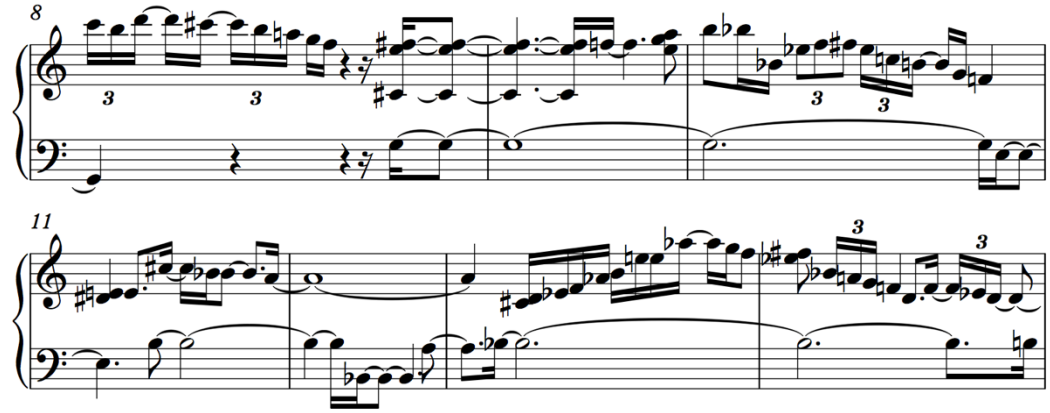


Figure 2.54: Bars 8-14 in the thirteenth 1 minute-output, fourth experiment

The subsequent passage (bars 32-35) is from the same output as the last one. Even though they are not consecutive in the output, it allows itself to start directly after the last fragment without interrupting the musical flow. The last chord in bar 34 is altered, which originally had a single different pitch from the previous chord in the same bar and caused harmonic stagnation.

More significant changes took place in the following passage (bars 36-44, originally extracted from bars 17-24 of the nineth 1 minute-output). The *sforzando* chords found in bars 37, 38 and 41 are all shifted an octave higher to introduce a novel idea to the passage. The climax of the miniature occurs in bar 41 of this passage as well, to which the mentioned idea contributes significantly. Other alterations are mainly done for voice leading purposes, like the insertion of D and G natural in the first beat of 39 and the push of the A#3 (originally A#2) in the third beat of bar 42 up an octave.

Lastly, the passage in bars 45-52 (taken from bars 2-8 of the seventh 1 minute-output, shown in Figure 2.55) is intended for providing a musical relief after the climactic moment in the piece. The pitch content is slightly altered compared to the version in the output, mostly for aesthetic preferences and avoiding exaggerated consonances. Last chords of the piece seen in the last four bars are composed manually.

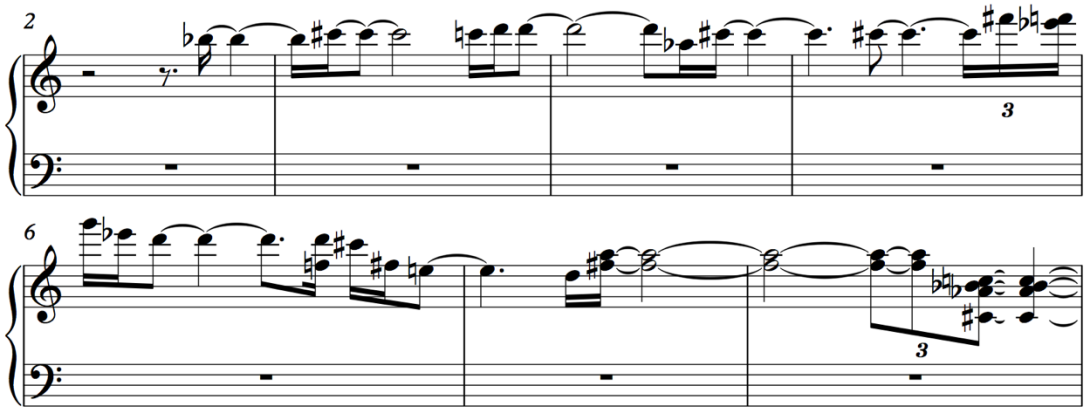


Figure 2.55: Bars 2-8 in the seventh 1 minute-output, fourth experiment

3. CONCLUSION AND DISCUSSION

In the first experiment, I trained Performance RNN with the entire piano repertoire of Olivier Messiaen. Some of his musical language was observed in the outputs, but the overall accuracy was quite low. To improve the accuracy in the next experiment, I added one more layer to the neural network to increase its learning capability and eliminated the velocity data of the input dataset. The outputs, however, turned out to be less musical compared to the first experiment. For the third experiment I decided to enlarge the dataset with the works of different composers. Four composers (Murail, Stockhausen, Ferneyhough and Schönberg, Webern and Berg as a single group) were chosen for this purpose out of the predetermined pool of eight composers with the help of another neural network model called Music SOM. This model clusters musical works according to the similarities in their spectral features. Performance RNN reacted successfully to the enlarged dataset; fragments in the outputs frequently demonstrated musical style comparable to those of the composers added to the dataset. In the last experiment I trained the neural network with my own improvisations. Then, I used the outputs of the model for composing a piano miniature, providing an example for the case of a neural network assisting a human composer in the context of contemporary music.

Looking back to the first two experiments where I tested the capabilities of a neural network outside of the idiom of tonal music, it is apparent that the model and training data used in this set were trying to overfit instead of generalizing the style of Messiaen. In spite of that, it learned the training data up to sixty percent accuracy and we were able to see some of Messiaen's musical language in the outputs. Because of the relatively low level of accuracy, we can claim that model generates original works loosely inspired by a subset of Messiaen's compositions.

As the ANN models for music generation improved drastically in the last few years, there should be more discussion about the chosen datasets, as they are the most crucial component on the way to more meaningful musical results. The experiments in this thesis showed us that when working with complex music, small datasets may be

insufficient; however, increasing the dataset size requires sacrificing from consistency of style, especially in 20th and 21st century music with many diverse styles of composition. Therefore, great care should be exercised for the decisions about the size and content of a dataset.

Depending on the purpose of the application at hand, different choices about the dataset can be made. Tonal works of the common practice period are obviously easier to work with for the sole purpose of style imitation, where datasets of larger sizes can be obtained by compiling the works of composers with a similar musical style. On the other hand, for the purposes of contemporary composition the fourth experiment in this thesis is more relevant. Of course, an important question in this context is where a composer places the material gathered from the outputs in the spectrum spreading from raw material to end product. The outputs generated by Performance RNN is appropriate for using as raw material but they are not suitable if a composer is trying to generate material as an end product, as RNNs do not generate formally coherent outputs. Music transformer by Huang et al. (2018) should be investigated for that purpose, as mentioned in the previous sections.

In order to get better results within the context of contemporary music, the ways of compiling a dataset must be discussed as well. The audio-to-midi transcription model used in this thesis performed worse with Messiaen than it performs with common practice period composers; a more accurate representation of the actual works would undoubtfully improve the results we obtain from any model. More meaningful experiments can be made if the neural network-based transcription models are trained with and improved for contemporary music, or manually created MIDI datasets of contemporary music become increasingly available in the future.

Lastly, different encoding types used by different models also have a significant effect on the outcome. Performance RNN states its purpose as creating performances and it encodes time information based on milliseconds, which arguably make the learning process for complicated rhythms more difficult. In other words, rhythmic complexities arising from performances may interfere with the learning of complex rhythms in compositions. Therefore, a duration encoding based on exact note divisions may be more efficient for the purpose of composition assistance, on the condition that tuplets can be encoded as well.

REFERENCES

Boulanger-Lewandowski, N. (2015). *Deep Learning Tutorial* [Ebook]. LISA Lab, University of Montreal. Retrieved October 21, 2019, from http://deeplearning.net/tutorial/deeplearning.pdf

Boulanger-Lewandowski, N., Bengio, Y., & Vincent, P. (2012). Modeling Temporal Dependencies in High-Dimensional Sequences: Application to Polyphonic Music Generation and Transcription. In *Proceedings of the 29th International Conference on Machine Learning,* Edinburgh, Scotland, UK.

Briot, J., Pachet, F., & Hadjeres, G. (2019). *Deep learning techniques for music generation – a survey.* Retrieved October 20, 2019, from https://arxiv.org/abs/1709.01620v4

Chandra, A. (2018). McCulloch-Pitts Neuron — Mankind's First Mathematical Model Of A Biological Neuron. *Towards Data Science.* Retrieved October 25, 2019, from https://towardsdatascience.com/mcculloch-pitts-model-5fdf65ac5dd1

Chandra, A. (2018). Perceptron: The Artificial Neuron (An Essential Upgrade To The McCulloch-Pitts Neuron). *Towards Data Science.* Retrieved October 25, 2019, from https://towardsdatascience.com/perceptron-the-artificial-neuron-4d8c70d5cc8d

Eck, D. (2002). Finding temporal structure in music: Blues improvisation with LSTM recurrent networks. In *Proceedings of the 12th IEEE workshop on neural networks for signal processing* (pp. 747-756). IEEE.

Emiya, V., Bertin, N., David, B., & Badeau, R. (2010). *MAPS - A piano database for multipitch estimation and automatic transcription of music* [Research Report]. Paris: Telecom ParisTech.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. Boston, Massachusetts: MIT Press.

Hadjeres, G., Pachet, F., & Nielsen, F. (2017). DeepBach: A Steerable Model for Bach Chorales Generation. In *Proceedings of the 34th International Conference on Machine Learning,* Sydney, Australia.

Harte, C., Sandler, M., & Gasser, M. (2006). Detecting Harmonic Change in Musical Audio. In *Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia* (pp. 21-26). Santa Barbara, California, USA: ACM Press.

Hawthorne, C., Elsen, E., Song, J., Roberts, A., Simon, I., & Raffel, C. et al. (2018). *Onsets and Frames: Dual-Objective Piano Transcription.* Retrieved October 27, 2019, from https://arxiv.org/abs/1710.11153

Hild, H., Feulner, J., & Menzel, W. (1991). HARMONET: a neural net for harmonizing chorales in the style of J.S. Bach. In *Proceedings of the 4th International Conference on Neural Information Processing Systems* (pp. 267-274). San Francisco, California, USA: Morgan Kaufmann Publishers Inc.

Hinton, G., Osindero, S., & Teh, Y. (2006). A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, *18*(7), 1527-1554.

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735-1780.

Huang, C., Aswani, A., Uszkoreit, J., Shazeer, N., Simon, I., & Hawthorne, C. et al. (2018). *Music Transformer.* Retrieved October 27, 2019, from https://arxiv.org/abs/1809.04281

Jiang, D., Lu, L., Zhang, H., & Tao, J. (2002). Music type classification by spectral contrast feature. In *Proceedings of the 2002 IEEE International Conference on Multimedia and Expo*. Lausanne, Switzerland: IEEE.

Johnson, D. (2015). *Composing Music with Recurrent Neural Networks*. Retrieved October 28, 2019, from http://www.hexahedria.com/2015/08/03/composing-music-with-recurrent-neural-networks/

Karpathy, A. (2015). The Unreasonable Effectiveness of Recurrent Neural Networks. *GitHub.* Retrieved October 28, 2019, from http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Khazri, A. (2019). Self Organizing Maps. *Towards Data Science.* Retrieved November 5, 2019, from https://towardsdatascience.com/self-organizing-maps-1b7d2a84e065

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, *43*(1), 59-69.

Kohonen, T., Huang, T., & Schroeder, M. (2012). *Self-Organizing Maps*. Berlin, Heidelberg: Springer Berlin / Heidelberg.

Krystalakos, O. (2017). music-som. *GitHub.*Retrieved November 6, 2019, from https://github.com/OdysseasKr/music-som

Kurenkov, A. (2015). *A 'Brief' History of Neural Nets and Deep Learning.* Retrieved October 25, 2019, from https://www.andreykurenkov.com/writing/ai/a-brief-history-of-neural-nets-and-deep-learning/

Mao, H., Shin, T., & Cottrell, G. (2018). DeepJ: Style-Specific Music Generation. In *Proceedings of 12th International Conference on Semantic Computing*. Laguna Hills, California, USA: IEEE.

McCulloch, W., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin Of Mathematical Biophysics*, *5*(4), 115-133.

McDonald, K. (2017). Neural Nets for Generating Music. *Medium.* Retrieved October 28, 2019, from https://medium.com/artists-and-machine-intelligence/neural-nets-for-generating-music-f46dffac21c0

McFee, B., Raffel, C., Liang, D., & Ellis, D. (2015). librosa: Audio and Music Signal Analysis in Python. In *Proceedings of the 14th Python in Science Conference, SciPy 2015* (pp. 18-24), Austin, Texas, USA.

Messiaen, O. (1956). *The technique of my musical language*. Paris: Alphonso Leduc.

Messiaen, O. (1958). *Catalogue d'Oiseaux.* Paris: Alphonso Leduc. (1964)

Metzer, D. (2011). *Musical modernism at the turn of the twenty-first century*. New York: Cambridge University Press.

Minsky, M., & Papert, S. (1969). *Perceptrons*. Cambridge: MIT Press.

Morgan, R. (1984). Secret Languages: The Roots of Musical Modernism. *Critical Inquiry*, *10*(3), 442-461. doi: 10.1086/448257

Nguyen, M. (2018). Illustrated Guide to LSTM's and GRU's: A step by step explanation. *Towards Data Science.* Retrieved October 28, 2019, from https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21

Nielsen, M. (2019). *Neural Networks and Deep Learning* [Ebook]. Retrieved October 25, 2019, from http://neuralnetworksanddeeplearning.com/chap2.html

Nierhaus, G. (2009). *Algorithmic Composition*. Dordrecht: Springer.

Oore, S., Simon, I., Dieleman, S., Eck, D., & Simonyan, K. (2018). *This time with feeling: learning expressive musical performance.* Retrieved October 27, 2019, from https://arxiv.org/abs/1808.03715

Roberts, A., Mann, Y., & Engel, J. (2019). Magenta Studio. *Magenta.* Retrieved October 23, 2019, from https://magenta.tensorflow.org/studio-announce

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386-408.

Rumelhart, D., Hinton, G., & Williams, R. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533-536.

Simon, I., Huang, C., & Dinculescu, M. (2018). Music Transformer: Generating Music with Long-Term Structure. *Magenta.* Retrieved November 2, 2019, from https://magenta.tensorflow.org/music-transformer

Stockhausen, K. (1961). *Klavierstück IX.* London: Universal Edition. (1967)

Straus, J. (2005). *Introduction to post-tonal theory* (3rd ed.). New Jersey, USA: Pearson.

Tjoa, S. (2015). Mel Frequency Cepstral Coefficients (MFCCs). *GitHub.* Retrieved November 8, 2019, from https://github.com/stevetjoa/musicinformationretrieval.com/blob/gh-pages/mfcc.ipynb

Todd, P. (1989). A Connectionist Approach to Algorithmic Composition. *Computer Music Journal*, *13*(4), 27.

Toop, R. (1990). Brian Ferneyhough's Lemma-Icon-Epigram. *Perspectives of New Music*, *28*(2), 52.

van den Oord, A., Kavukcuoglu, K., Zen, H., Dieleman, S., Simonyan, K., & Vinyals, O. et al. (2016). *WaveNet: A Generative Model for Raw Audio.* Retrieved October 25, 2019, from https://arxiv.org/abs/1609.03499

Url-1 <https://librosa.github.io/librosa/>, date retrieved 06.09 2019.

APPENDICES

APPENDIX A: Commands for running Onsets and Frames locally
APPENDIX B: First two pages of the longer output of the first experiment
APPENDIX C: Code fragment of Performance RNN for the Performance with the 'performance_with_dynamics_compact' configuration
APPENDIX D: First two pages of the longer output of the second experiment
APPENDIX E: Calls for the training function of Music SOM
APPENDIX F: First two pages of the longer output of the third experiment
APPENDIX G: Sample of improvisation transcription
APPENDIX H: First two pages of the longer output of the fourth experiment
APPENDIX I: Comprovisational miniature

APPENDIX A: Commands for running Onsets and Frames locally

```
MODEL_DIR=<path to directory containing checkpoint>
onsets_frames_transcription_transcribe \
  --model_dir="${CHECKPOINT_DIR}" \
  <piano_recording1.wav, piano_recording2.wav, ...>
```

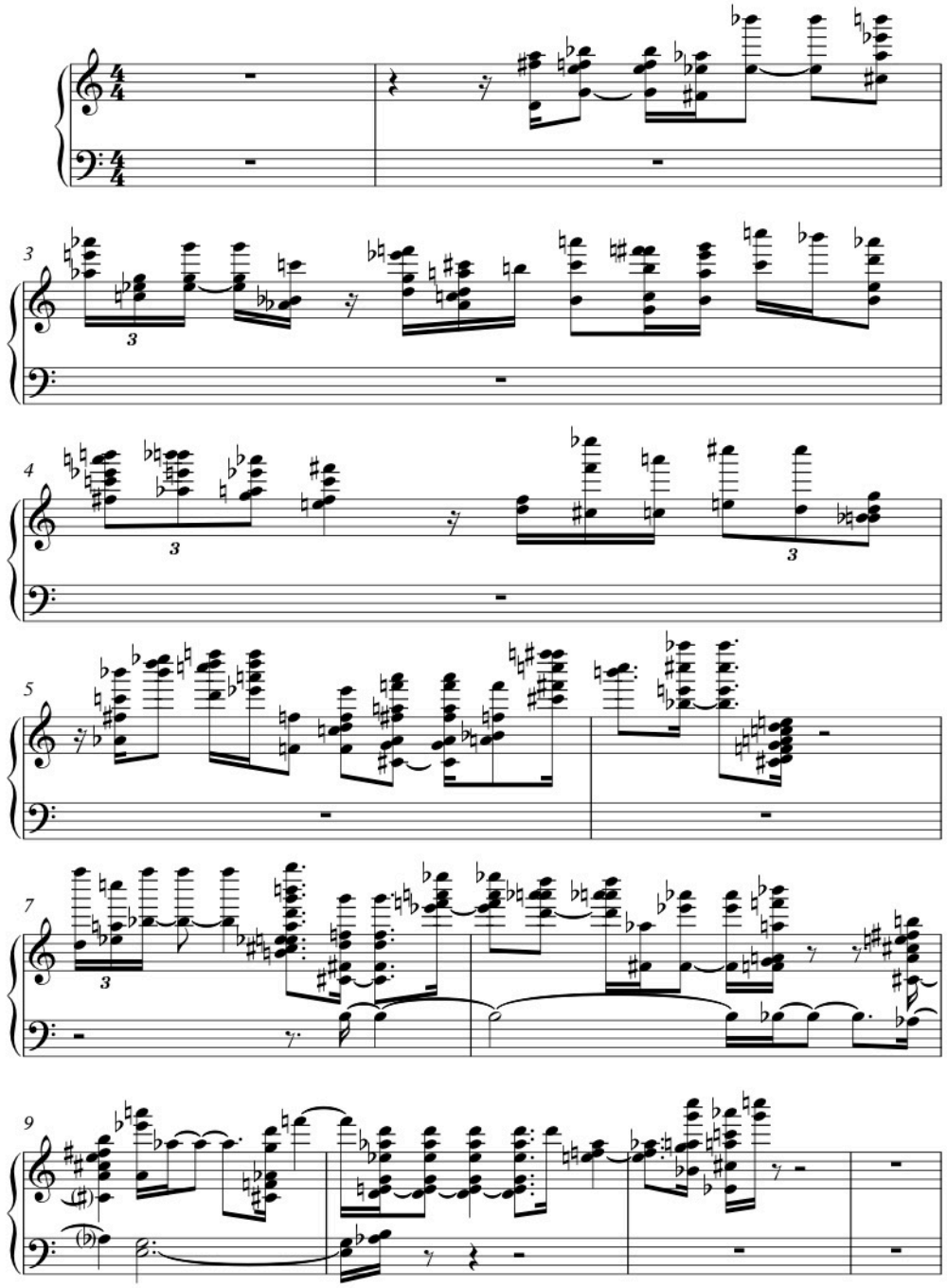
Figure A.1: Original code for running Onsets and Frames locally[29]

```
MODEL_DIR=<path to directory containing checkpoint>
onsets_frames_transcription_transcribe \
  --model_dir=${MODEL_DIR} \
  <piano_recording1.wav, piano_recording2.wav, ...>
```

Figure A.2: Corrected code for running Onsets and Frames locally

[29] On 06.11.2019, retrieved from https://github.com/tensorflow/magenta/tree/master/magenta/models/onsets_frames_transcription

APPENDIX B: First two pages of the longer output of the first experiment



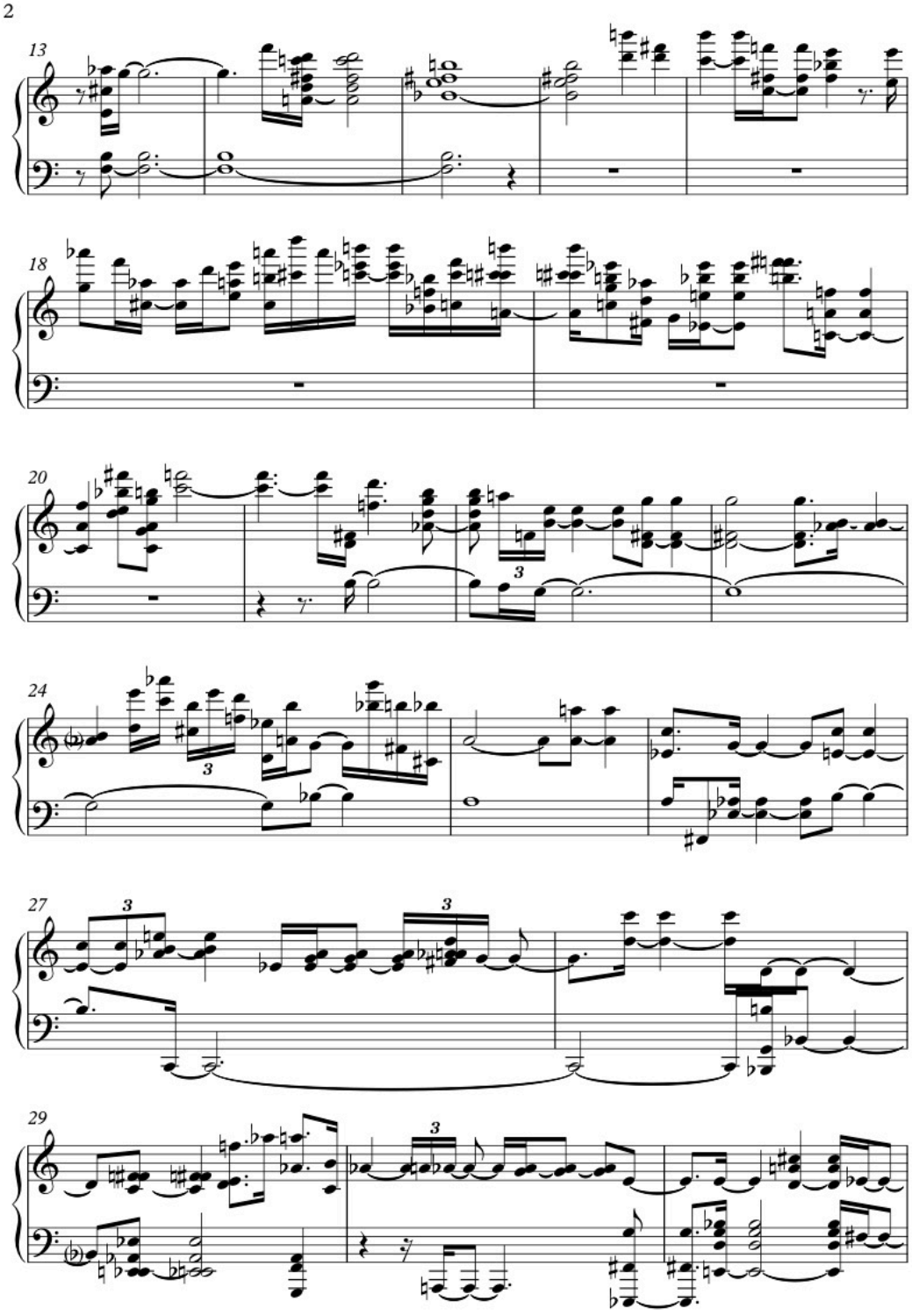


Figure A.3: Sample output from the first experiment

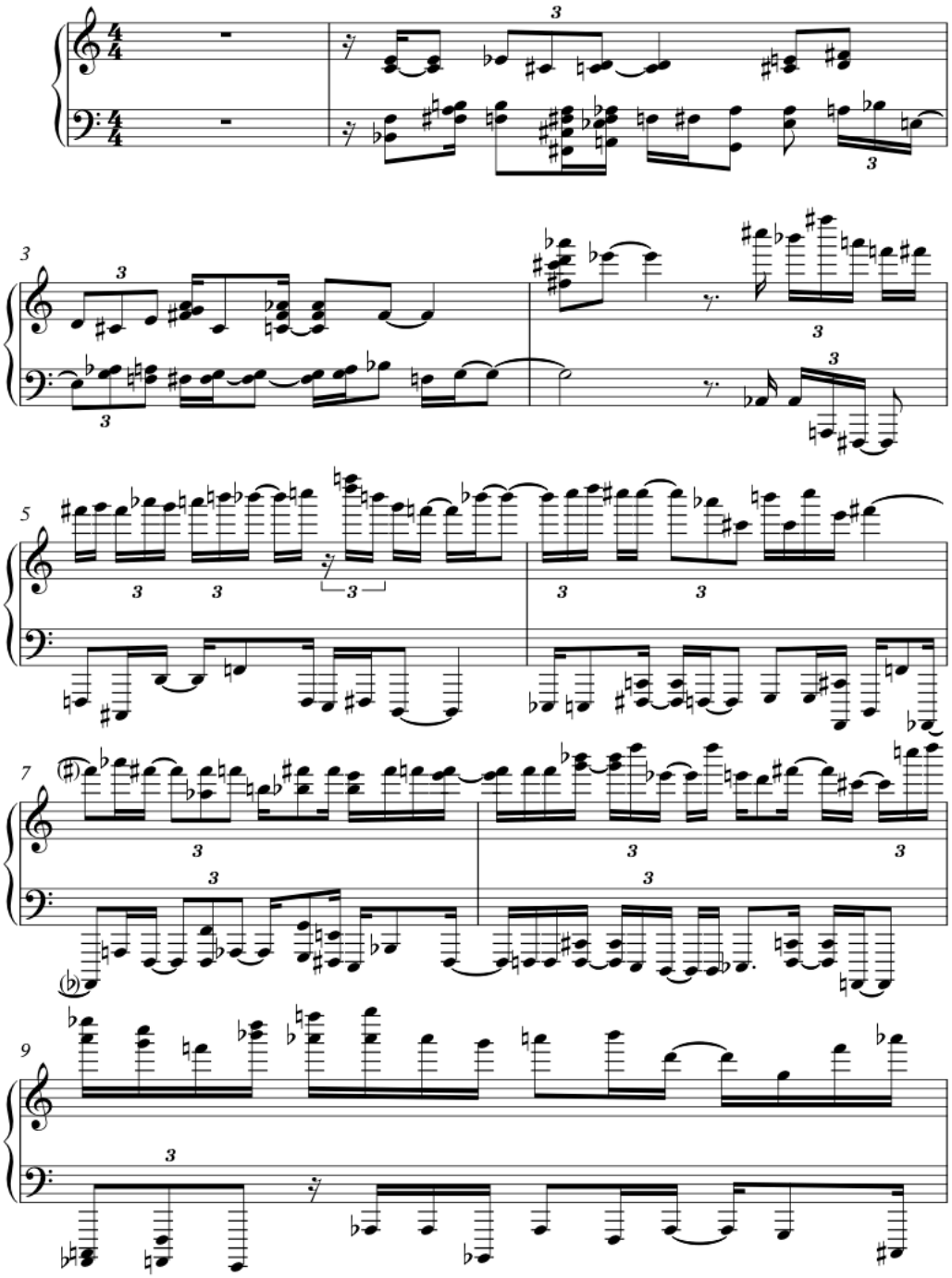
APPENDIX C: Code fragment of Performance RNN for the Performance with the 'performance_with_dynamics_compact' configuration

```
'performance_with_dynamics_compact': PerformanceRnnConfig(
        magenta.protobuf.generator_pb2.GeneratorDetails(
            id='performance_with_dynamics',
            description='Performance RNN with dynamics (compact
            input)'),
        magenta.music.OneHotIndexEventSequenceEncoderDecoder(
            magenta.music.PerformanceOneHotEncoding(
                num_velocity_bins=32)),
        tf.contrib.training.HParams(
            batch_size=64,
            rnn_layer_sizes=[512, 512, 512],
            dropout_keep_prob=1.0,
            clip_norm=3,
            learning_rate=0.001),
        num_velocity_bins=32)
```

Figure A.4: 'performance_with_dynamics_compact' configuration[30]

[30] num_velocity_bins is changed to 1 and rnn_layer_sizes is changed to [512, 512, 512, 512] in the second experiment.

APPENDIX D: First two pages of the longer output of the second experiment



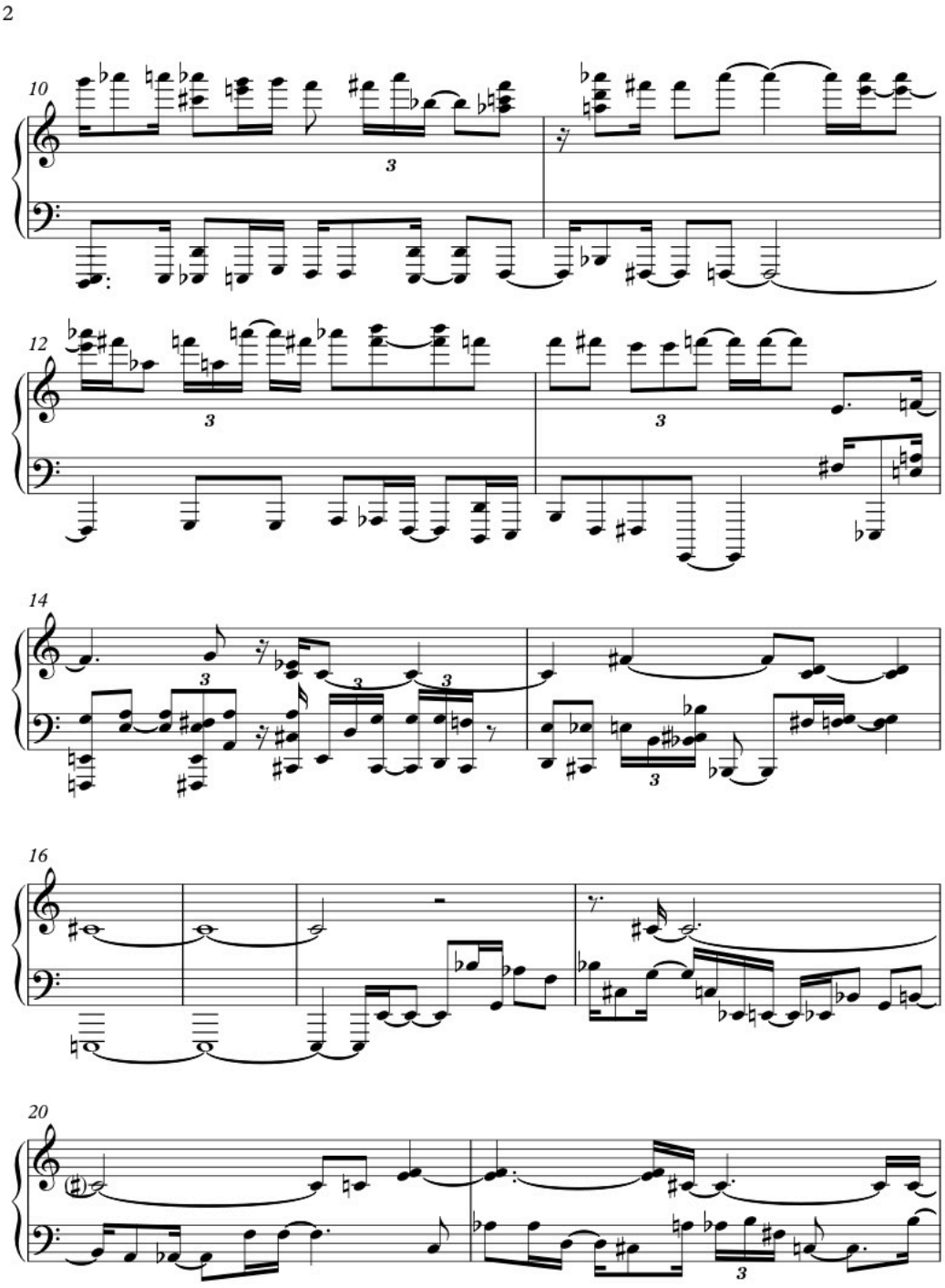


Figure A.5: Sample output from the second experiment

APPENDIX E: Calls for the training function of Music SOM

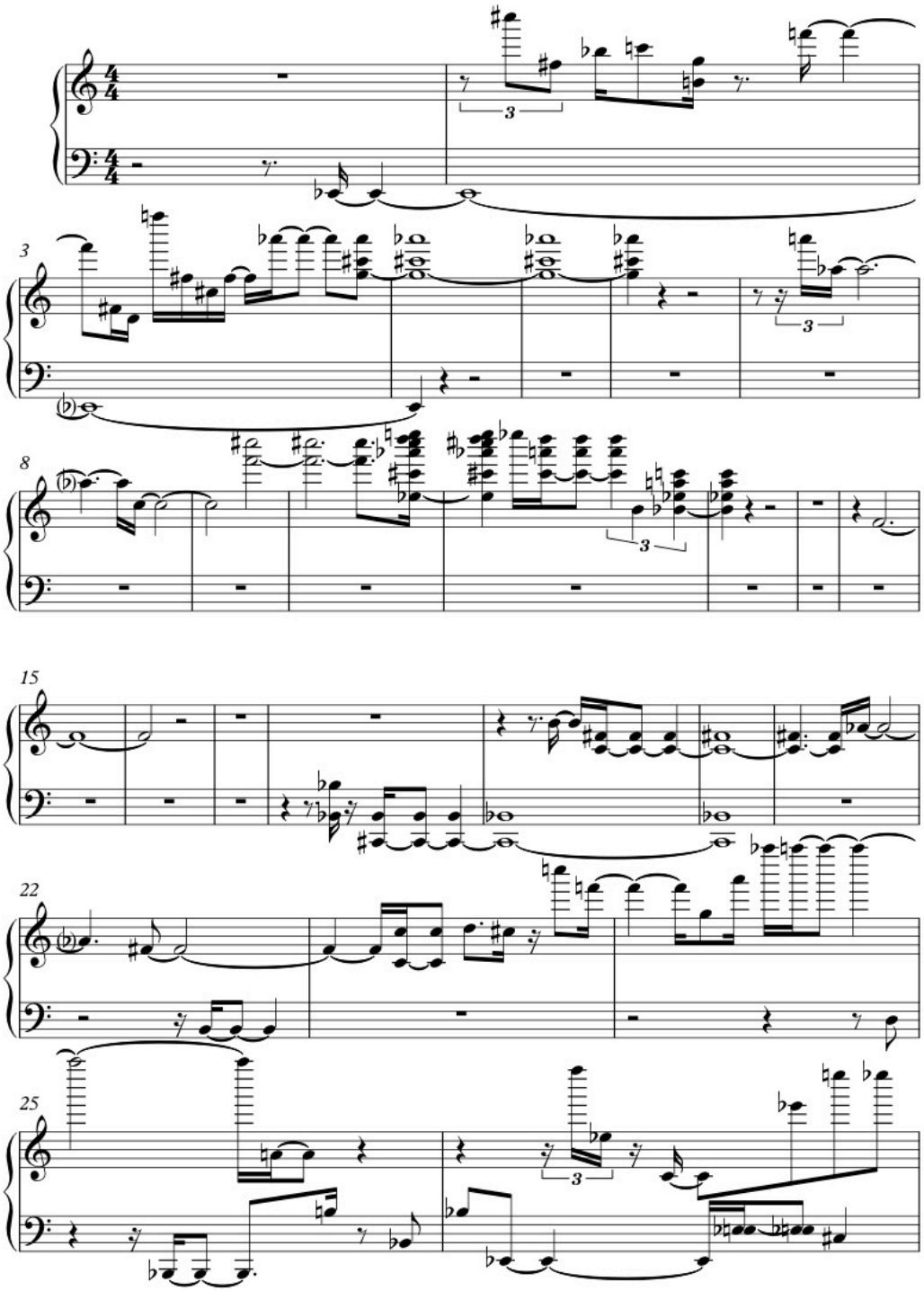
```
somap.train(train_samples[:21],train_labels[:21],1,3)
somap.train(train_samples[21:31],train_labels[21:31],0.5,3)
somap.train(train_samples[31:41],train_labels[31:41],0.1,1)
somap.train(train_samples[41:],train_labels[41:],0.1,0.1)
```

Figure A.6: Original code for calling the training function of Music SOM

```
for x in range(250):
    somap.train(train_samples[:31],train_labels[:31],1,3)
    train_samples, train_labels =
unison_shuffled_copies(train_samples, train_labels)

for x in range(250):
   somap.train(train_samples[31:41],train_labels[31:41],0.5,3)
    train_samples, train_labels =
unison_shuffled_copies(train_samples, train_labels)

for x in range(250):
   somap.train(train_samples[41:51],train_labels[41:51],0.1,1)
    train_samples, train_labels =
unison_shuffled_copies(train_samples, train_labels)

for x in range(250):
    somap.train(train_samples[51:],train_labels[51:],0.1,0.1)
    train_samples, train_labels =
unison_shuffled_copies(train_samples, train_labels)
```

Figure A.7: Updated code for calling the training function of Music SOM

APPENDIX F: First two pages of the longer output of the third experiment



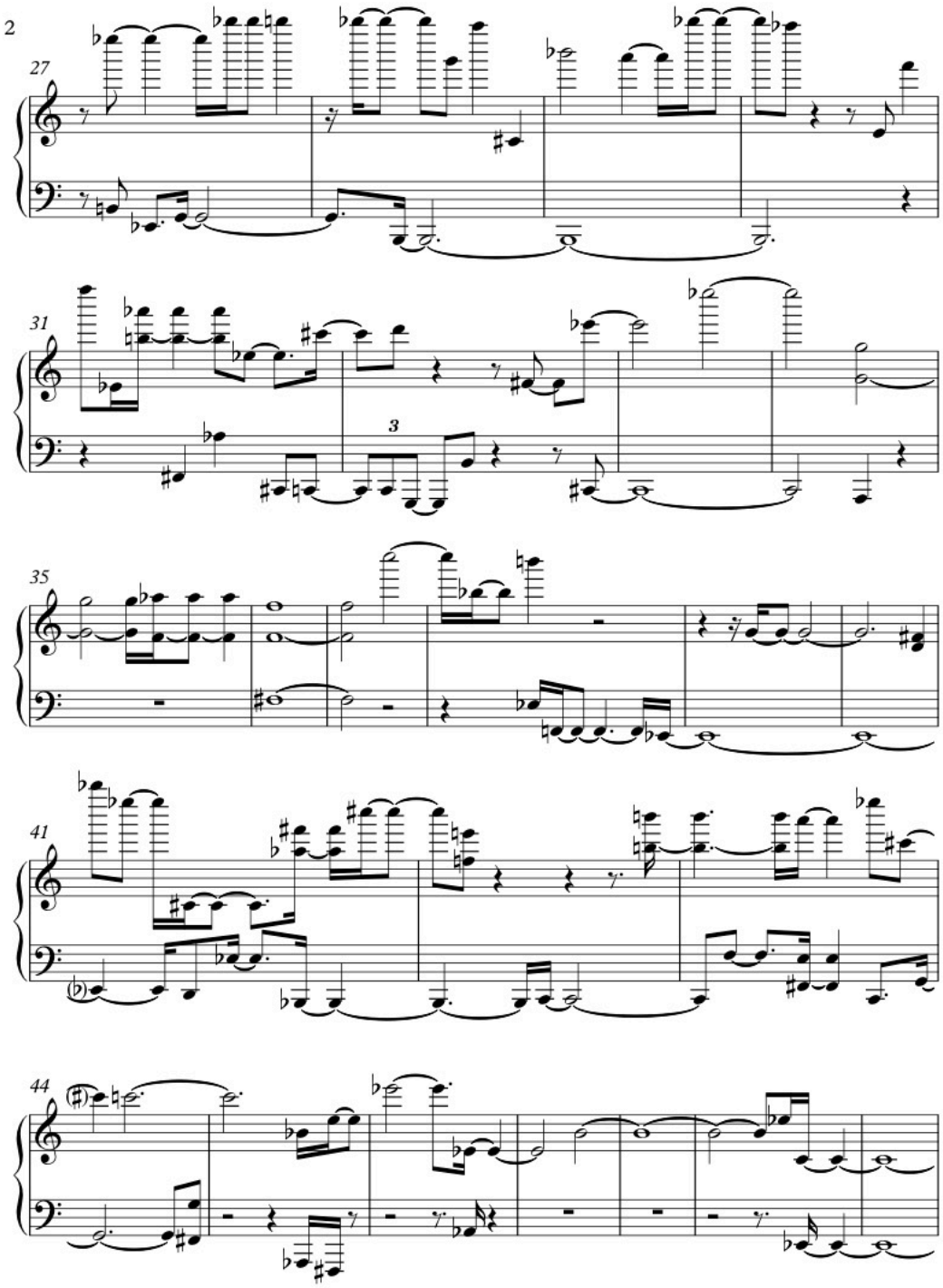
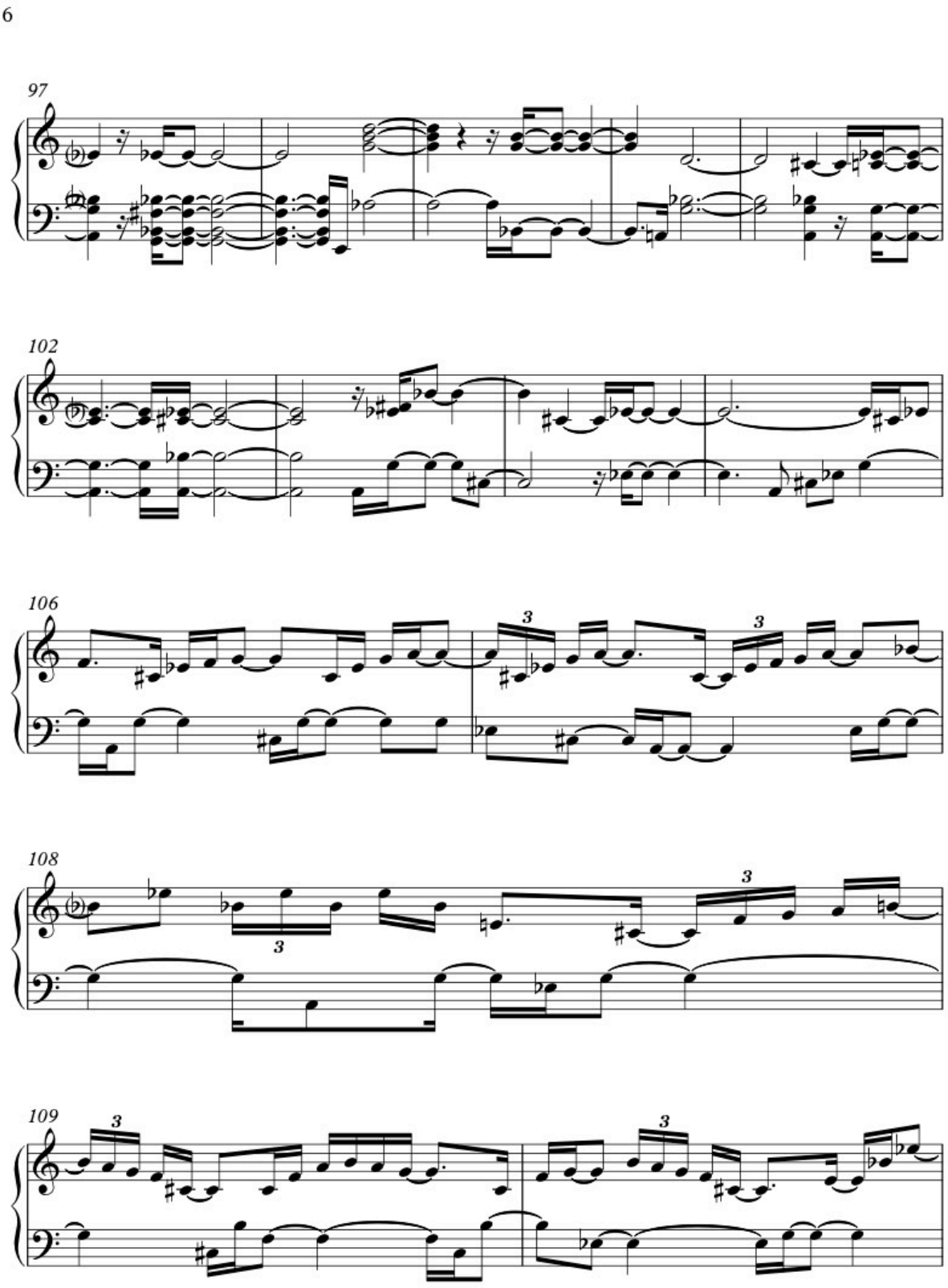


Figure A.8: Sample output from the third experiment

APPENDIX G: Sample of improvisation transcription



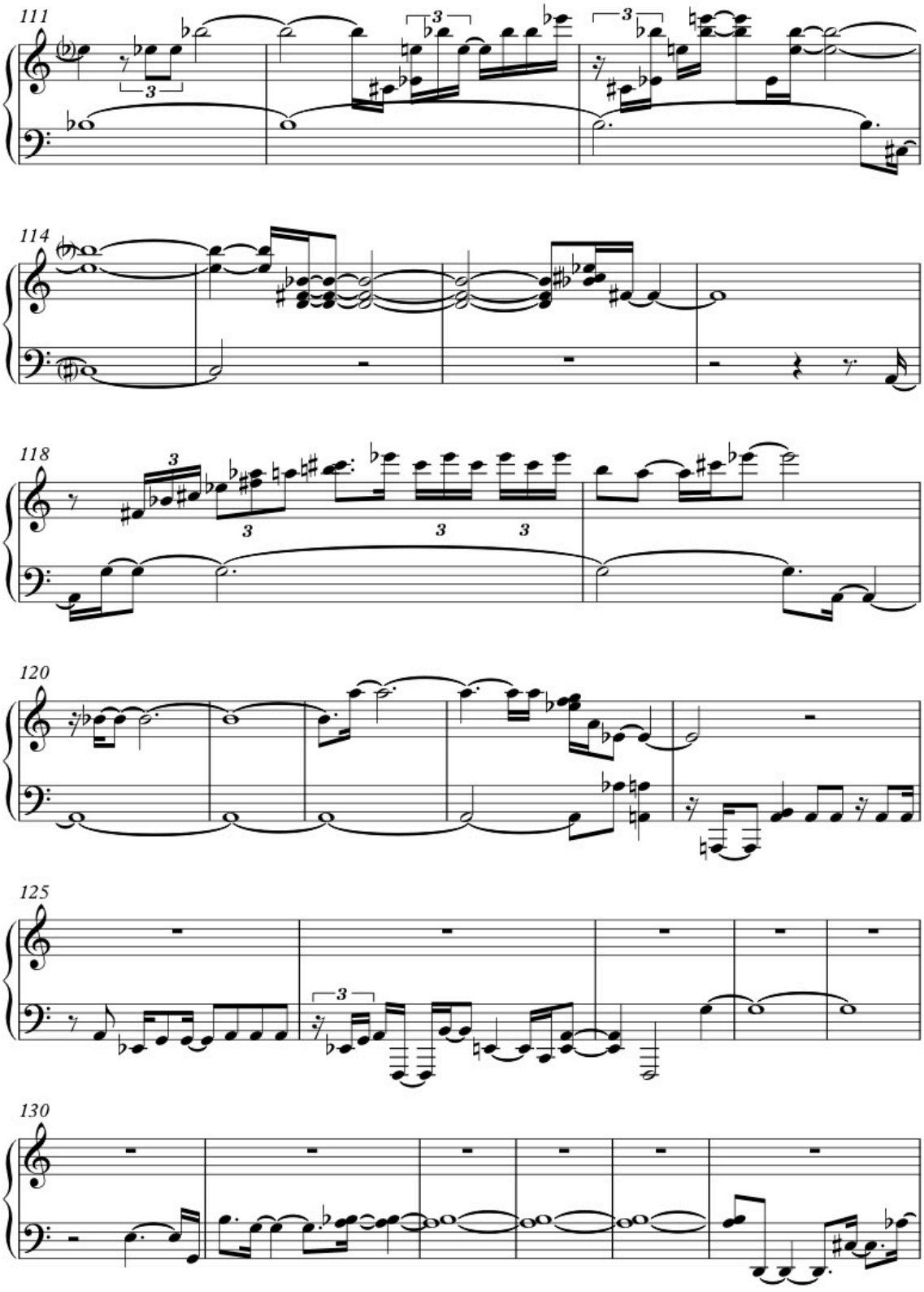
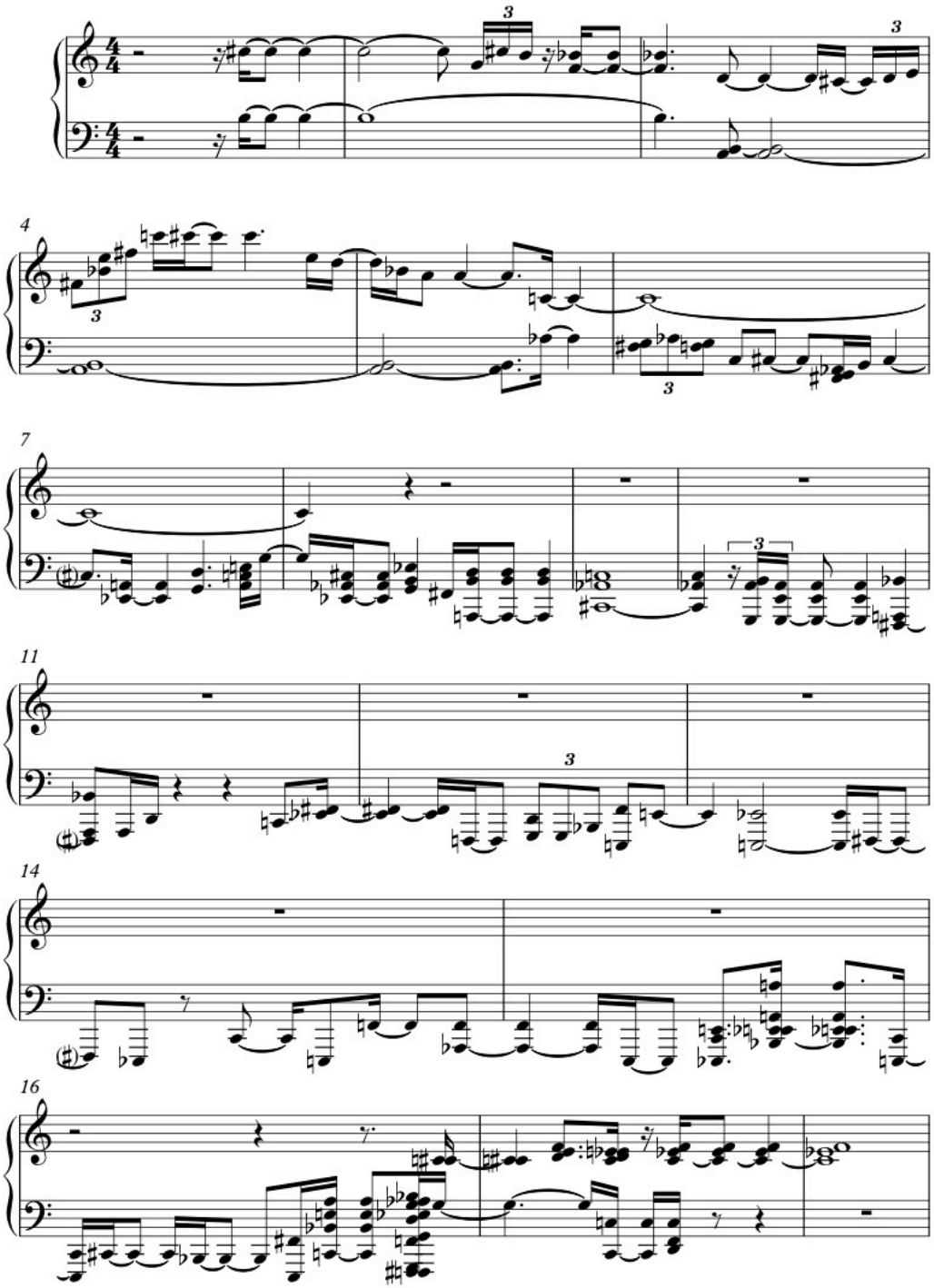


Figure A.9: Sample of improvisation transcription

APPENDIX H: First two pages of the longer output of the fourth experiment



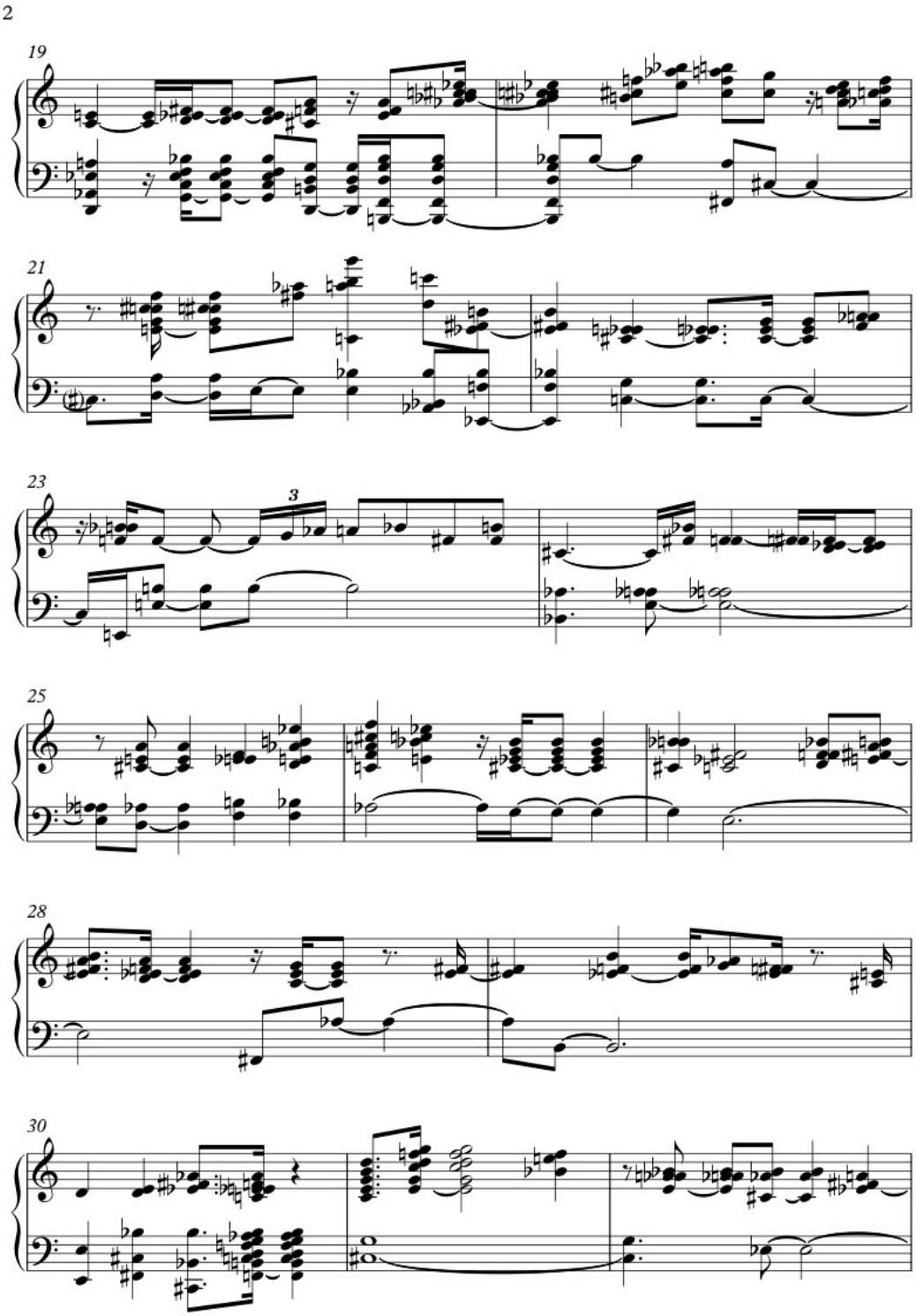
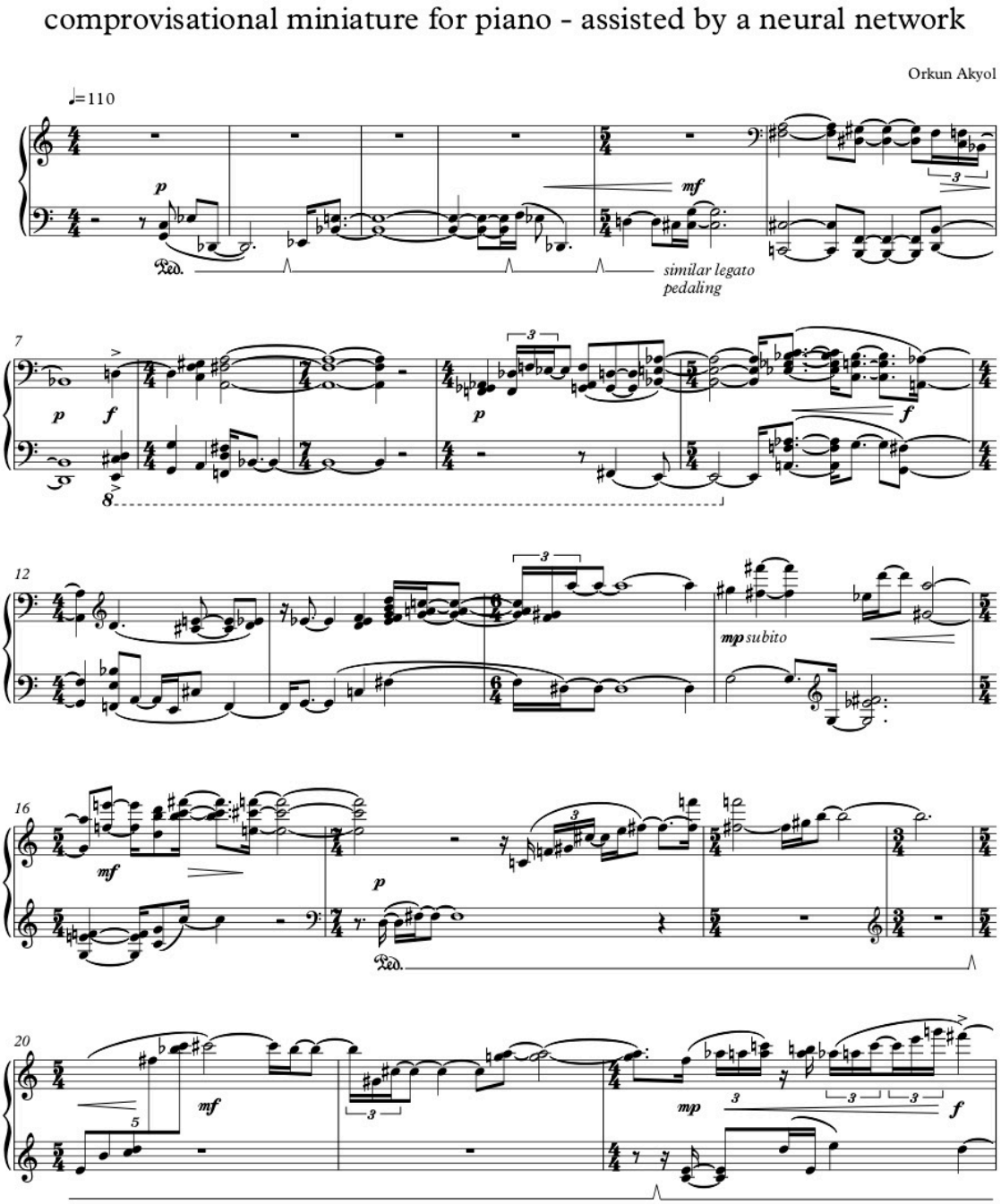
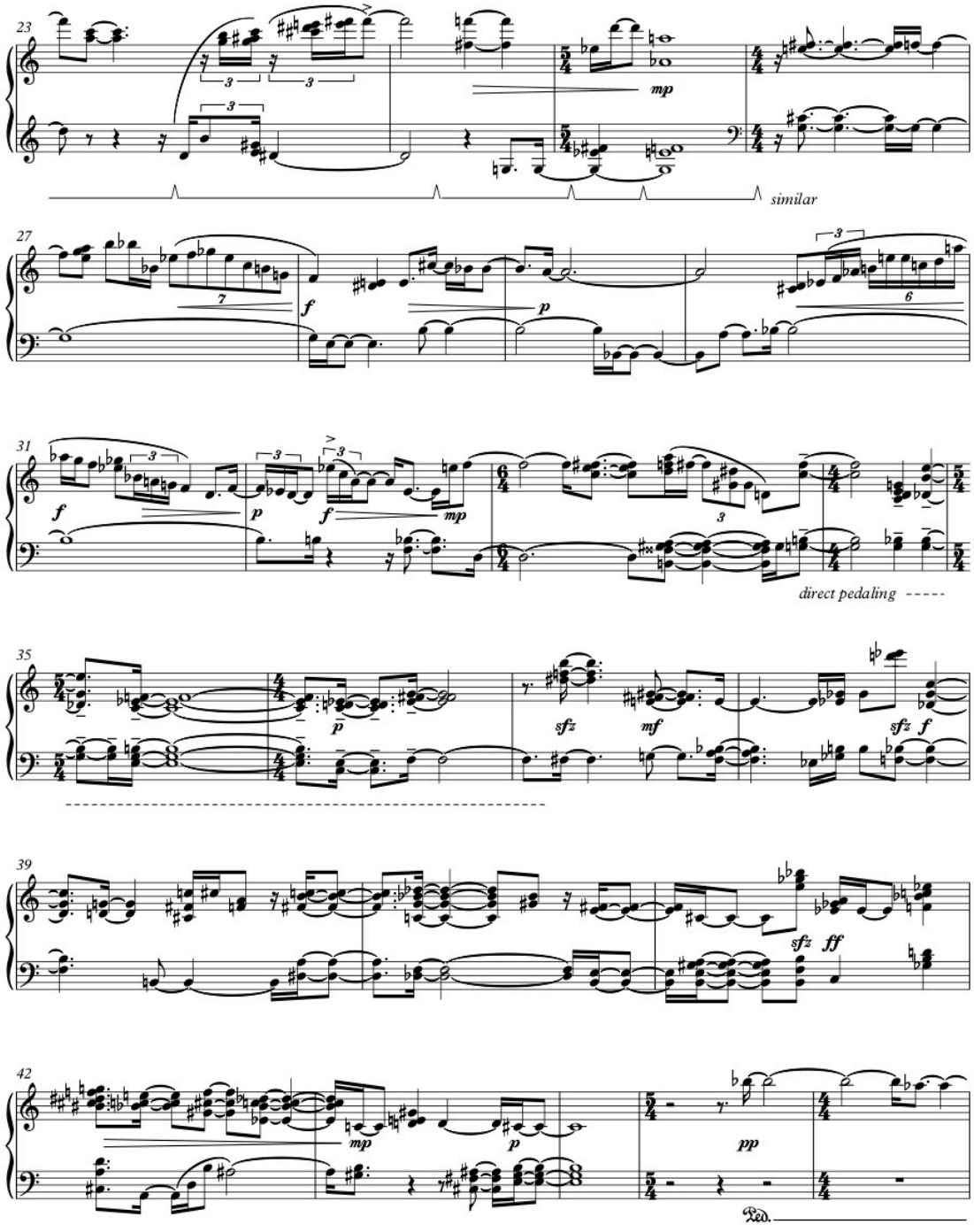


Figure A.10: Sample output from the fourth experiment

APPENDIX I: Comprovisational miniature




23
mp
similar
27
f
p
31
f
p
f
mp
direct pedaling
35
p
sfz
mf
sfz
f
39
sfz
ff
42
mp
p
pp
Ped.

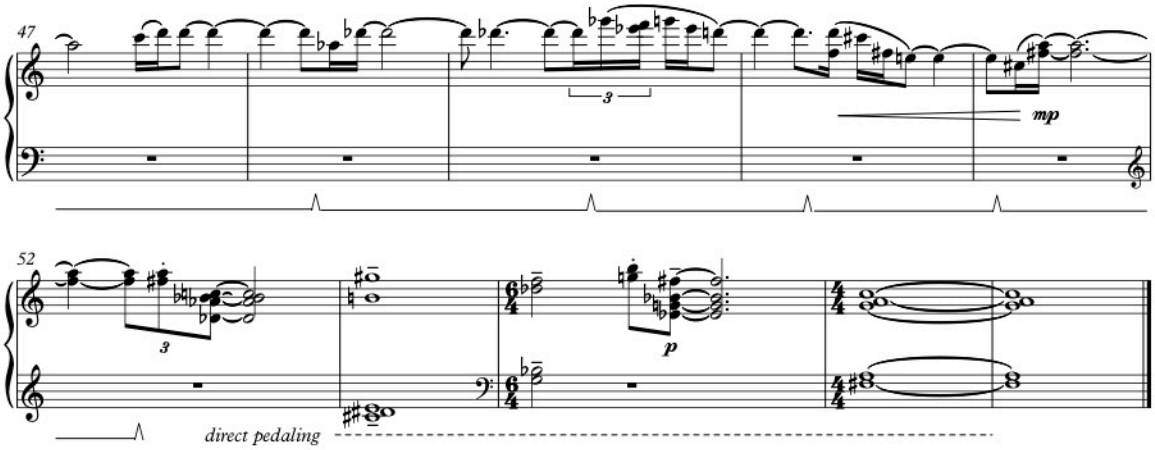


Figure A.11: Comprovisational miniature

CURRICULUM VITAE

Name Surname	**:** Tuğrul Orkun Akyol
Place and Date of Birth	**:** Kadıköy, İstanbul, 23.04.1992
E-Mail	**:** akyolorkun@gmail.com
Bachelor of Science	**:** 2016, Boğaziçi University, Engineering Departmant, Industrial Engineering