TURKISH MAKAM MUSIC GENERATION USING TRANSFORMERS

by

Ahmet Taha Bilgiç

B.S., Electrical and Electronics Engineering, Boğaziçi University, 2018

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering
Boğaziçi University

2023

ACKNOWLEDGEMENTS

I am deeply grateful to my family for their unwavering support throughout my educational journey and for fostering my interest in music. I would like to extend my heartfelt thanks to my friends, particularly Azad Yaşar, my older colleague and classmate from graduate courses who introduced me to the field of Transformers.

I am deeply indebted to my thesis advisor, Assistant Professor Berk Gökberk, and my previous advisor, Professor Haluk Bingöl, for their guidance, support, and patience. Without their mentorship, I would not have been able to complete my Master's degree.

I also extend my sincere thanks to the members of my thesis jury, Associate Professor Arzucan Özgür and Assistant Professor Furkan Kıraç, for their valuable feedback and insights.

I am also grateful to the musicians who participated in the evaluation survey for their valuable contributions.

Lastly, I am forever grateful to Fikret Karakaya, my master, a highly respected musician and musicologist, who taught me most of what I know about music. The data used in this study was sourced from his archive and throughout the study, he provided me with invaluable support and guidance in the musical aspect.

ABSTRACT

TURKISH MAKAM MUSIC GENERATION USING TRANSFORMERS

With advancement of generative models and structures like transformers mostly used in natural language processing, development in music generation has been accelerated. However music generation studies are focused on tonal, Western music. In this study, we use transformers for melody generation to Makam Music, a traditional genre of monophonic music from the Middle East and Balkans. The proposed method involves training a transformer model on preprocessed and melody-extracted data from real performances, and finetuning the model for generation of "taksim", a musical form related to improvisation in specific makams. We conclude with generated plausible melodies in chosen makam which resembles taksim form with some dynamics. This work highlights the potential of deep learning techniques for the generation of non-Western musical styles.

ÖZET

DÖNÜŞTÜRÜCÜLER KULLANARAK TÜRK MAKAM MÜZİĞİ ÜRETİMİ

Üretken modellerin ve yoğun olarak doğal dil işlemede kullanılan dönüştürücüler gibi yapıların diğer alanlara olduğu gibi müzik üretiminde kullanılmasıyla bu alanda da önemli bir ilerleme yaşanmıştır. Ancak müzik üretimi çalışmaları genellikle tonal müzikte ve Batı müziğinin altında yoğunlaşmıştır. Bu çalışmada Orta Doğu ve Balkanlar'ın tek sesli bir müzik türü olan Makam Müziği'nde ezgi üretimi için dönüştürücüler kullanılmıştır. Gerçek icraların ses dosyaları işlenerek elde edilen veriler üzerinde bir dönüştürücü modeli eğitilmiş ve belirli makamlarda doğaçlama ile ilgili bir müzik formu olan "taksim" üreten bir sistem amaçlanmıştır. Çalışma sembolik melodi üretimine odaklanmış olup, melodik sekans olarak taksim formunun dinamiklerine uygun sonuçlar elde edilmiştir. Bu çalışma, derin öğrenme tekniklerinin Batı müziği dışındaki tarzların üretilmesindeki potansiyelini ve teoriden bağımsız müzik üretimini ortaya koymaktadır.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii
ABSTRACT iv
ÖZET v
LIST OF FIGURES viii
LIST OF TABLES ix
LIST OF SYMBOLS x
LIST OF ACRONYMS/ABBREVIATIONS xi
1. INTRODUCTION 1
2. BASIC THEORY INFORMATION ABOUT MAKAM MUSIC 3
3. LITERATURE REVIEW 6
3.1. Music Generation Studies 6
3.1.1. Recurrent Neural Networks 7
3.1.2. Variational Auto-enconders 8
3.1.3. Generative Adversarial Networks 9
3.2. Makam Music Related Studies 10
3.3. Transformers and Music Generation 11
3.4. Overall Comment on Literature and Contributions 12
4. BACKGROUND 13
4.1. Self-Attention Mechanism and Transformers 13
4.2. Relative-Attention and Music Transformer 14
5. PROPOSED METHOD 16
5.1. Dataset 16
5.2. Preprocessing 17
5.2.1. Resampling 20
5.2.2. Filtering 20
5.2.3. True-silence Detection 21
5.2.4. Melody Extraction 21
5.2.5. Tokenization 23

5.2.6. False Silence Removal 24
5.2.7. Octave Error Removal 24
5.2.7.1. Quantitative Analysis of Octave Error Removal 25
5.3. Training 27
5.4. Generation 28
6. EXPERIMENT AND RESULTS 31
7. CONCLUSION & FUTURE WORK 34
REFERENCES 36

LIST OF FIGURES

Figure 2.1. Comparison between Tonal Music and Makam Music intervals from fretboard of Tanbur and Guitar 4

Figure 5.1. Overall system diagram 16

Figure 5.2. System diagram of preprocessing 19

Figure 5.3. Frequency response of Lowpass Butterworth filter 20

Figure 5.4. Before (above) and after (below) octave error and false silence removal example on a Taksim 26

LIST OF TABLES

Table 5.1. Octave error removal statistics on the sample set. 27

Table 6.1. Human evaluation Results 32

LIST OF SYMBOLS

f_0	Fundamental frequency

LIST OF ACRONYMS/ABBREVIATIONS

RNN	Recurrent Neural Network
GAN	Generative Adversarial Network
VAE	Variational Autoencoder
NLP	Natural Language Processing
MIDI	Musical Instrument Digital Interface
LSTM	Long-Short Term Memory
GPT	Generative Pre-Trained Transformer
FFT	Fast Fourier Transform
STFT	Short-Time Fourier Transform
VMD	Variational Mode Decomposition

1. INTRODUCTION

Computer-based music generation has been a continuously developing field of artificial intelligence since the first studies in the late 1960s. Different methods have been utilized in numerous studies, including but not limited to theory-driven rule-based methods, probabilistic models, evolutionary methods, and random number generation. With the exponential growth of data and the increasing capability of hardware, deep learning has become the most commonly used method. Due to the time-sequential nature of music, Recurrent Neural Networks (RNNs) were initially utilized in music generation systems, followed by more complex systems using RNNs, such as Variational autoencoders (VAE) and Generative Adversarial Networks (GAN). With the development of the attention mechanism and transformers for natural language processing (NLP), these have not only been utilized by NLP researchers for various problems, but have also been applied to other fields, including music generation. An example of a successful application of transformers in this field is the Music Transformer by Huang et al. [1].

Music generation studies can be grouped according to their focus. The objective may be to generate symbolic music in the form of a MIDI file or piano roll, directly raw audio signal with realistic timbres, a melody in a symbolic or sub-symbolic way, or an accompaniment for a selected melody. The emergence of end-to-end raw audio generation studies has not caused the extinction of other studies, as the focused and improving features can be different in each study.

Another direction in music generation is genre. Most studies are done in classical Western music and pop music, or more generally, tonal music. The main features of these genres are tonality, polyphony, and harmony. There are different classical music genres traditionally belonging to different geographic regions that are not Westernized or globalized. Makam music is a traditional music centered in the Middle East and Balkans that uses microtones and modes differently from tonal music. For this reason,

it is not suitable for polyphony and harmony, and is a melody-oriented monophonic music. While there are numerous music generation studies in Western classical music or universal pop music, Makam music generation is not as popular in music generation studies.

In Makam Music, or specifically Classical Turkish Music that we focus on this study, performance is the primary method of transmission. In the past, while classical European composers would write the notes of a new composition, classical Turkish composers would perform it for their patron or community. Although some notation systems were developed over the centuries, there was no consensus for notation in Turkish Makam music. In the 20th century, the Western notation system was adapted with additional symbols, but it is still not fully sufficient due to the complex grammatical rules of the makams. Therefore, we argue that real performances should be used in Makam music generation studies.

Additionally, in Makam music, there is a musical form closely related to performance called Taksim. Taksim is an improvisational performance on a specific makam by one or multiple instruments, used to prepare the listener for a makam, summarize a makam, and showcase instrumental skills.

In this study, we train a transformer with relative attention, similar to the Music Transformer. However, our data is preprocessed and melody-extracted from distinguished classical Turkish music performances. The data is not in MIDI or MIDI-like format, as is commonly used in the literature, but is instead tokenized by quantizing the pitch frequency scale with intervals at the level of the average human ear's distinguishing capacity. After training on the whole data set, we fine-tune the model for Taksim generation in three specific makams.

2. BASIC THEORY INFORMATION ABOUT MAKAM MUSIC

In 12-tone equal temperament system which is utilized by most todays music (i.e. Western music, Classical Western Music of last centuries), an octave is divided into 12 logarithmically equal pieces which are semitones. Accidentals (flats and sharps) divides a whole step logarithmically equally too. This system makes possible harmony which is the key point of Classical Western Music. On the other hand, Classical Turkish Music is monophonic and melody-oriented music which uses Pythagorean Scale in general, with involving several microtones. Number of microtones varies in different theories from about 50s to 70s per octave. Additionally, the main characteristic of this music is modal melodic system called "Makam". As the pitch intervals are not equal, their places can also change according to the Makam or even melodic progression in same Makam.

Mentioned difference can be seen in Figure 2.1 from the difference between fretboard of a Classical Turkish Music instrument, tanbur, and a guitar.

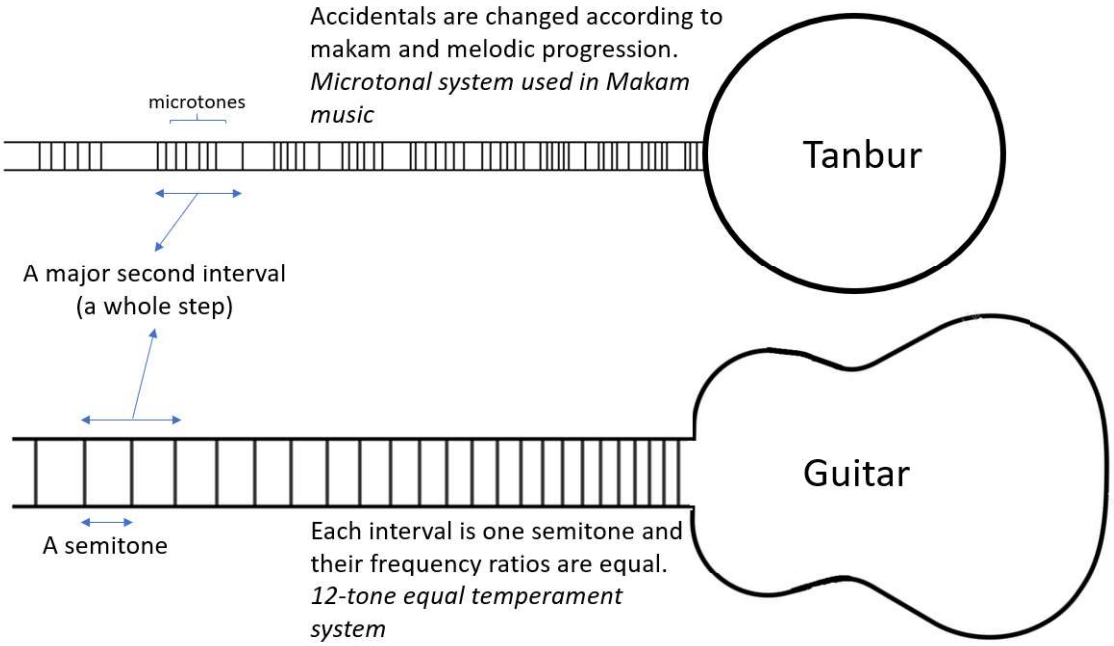


Figure 2.1. Comparison between Tonal Music and Makam Music intervals from fretboard of Tanbur and Guitar

Makams are constructed by attaching "cins" together. Cins is the melodic units of makams which can be trichords, tetrachords or pentachords. Cins' are used to construct the main modal scale of makams, and used for modulation and continuity between them. Even if modal scale is described for some makams with these cins', these are not enough to describe a makam. Makams are generally considered as melodic progression patterns rather than just a modal scale like popular modal scales in modal music, which are a subset of tonality. These conventional melodic progression patterns are called "Seyir". All makam-musical forms start melodic progression with the main makam and show the main characteristics in terms of scale and melody. Then they jump to different makams, which is called modulations, according to seyir rules of the main makam. Finally, the melody returns to the ending characteristics of the main makam and concludes with "karar" pitch, the tonal of the makam.

Taksim is an improvisational instrumental form in Makam Music generally having no rhythm. It can be performed by any instrument in any makam to describe the

makam characteristics and show instrumental skills. Like all Makam music forms, it starts with the makam which is selected, progresses with modulations to different makams, and brings to end with karar.

3. LITERATURE REVIEW

Music generation is one of the connection between artificial intelligence (AI) and art. There are numerous studies in AI-based music generation specialized on different genres [2–5], using different techniques like LSTM [2, 6, 7], VAE [8–10], transformers [1, 11], GANs [12] etc. or using and generating data in different information layer like raw audio data or symbolic representation [1, 13].

In the next sections, the related studies, some of which have been mentioned above will be summarized. In the first section, some important studies on music generation task using AI, then, related studies on Makam music will be elaborated second section. Because the center of this study is transformers and attention, the background studies will be detailed in the third section. Finally, the relation of this study with others and its novelty will be described.

3.1. Music Generation Studies

The first computer-based music generation was developed in 1957 [14] by Max Mathews, an electrical engineer at Bells Laboratory, and first algorithmic composition study utilizing Markov models has been started same year, and composed "Iliac Suite" for string quartet [15]. Since those days, music generation has been a growing field with several techniques utilized. Lots of method other than deep learning has been applied, Markov Model, Genetic Algorithm etc. However, with the acceleration of deep learning in last decades, most music generation application has been shifted to this area. Our focus is also deep learning methods and we detail mostly utilizing techniques in the studies.

3.1.1. Recurrent Neural Networks

Recurrent neural networks (RNN), differently from feed-forward networks, are ones the nodes have cyclic connections to allow feedback mechanism between output and input. Practically this recurrence mechanism brings ability to learn the state transition of time series environments.

Due to this time-sequential nature, recurrent neural network approach is one of the most used deep learning-based techniques. A very first deep learning-based melody generation study by Chen et al. [16] used RNN and genetic algorithm together for symbolic melody generation. Using genetic algorithm, they optimized neural network output with constraints describing the style of a composer and theoretical musical rules. With these rules they both resembled Bela Bartok's music and improved music quality according to diversity of pitch and rhythm. These constraints are used as fitness criteria of genetic algorithm and evolve RNN model.

Eck and Schimdhuber introduce Long-Short Term Memory RNNs (LSTM) to music in 2002 [3, 6] and they argue that LSTM advanced other RNNs in this task as "others cannot keep track of temporally distant events that indicate global music structure". In this study LSTM learns chord for given melody and also learns as chord-melody pairs in two different experiment.

Liang, in 2016, proposed a generative system based on LSTM to compose music in similar style particular composer or a selected genre [2]. As experiment, they feed the model with John Sebastian Bach's Chorales and analysed the results via musical Turing Test with online voluntary surveys. Results passed the test at the rate of above 90%.

Similarly, Huang et al. [17] trained LSTM network with different dataset such as one dataset including only Bach music and another one with several Classical music. The point we want to mention here is their evaluation. They applied a survey to 26

volunteers and asked them to rate the results on a scale from 1 to 10 as 1 is noise, 5 is plausible music and 10 is "composed by a novice composer".

3.1.2. Variational Auto-enconders

Autoencoder ıs a neural network consisting of two parts, encoder which compresses the data to a latent space and decoder which decompresses it again to higher dimensional space. Then autoencoders provides efficient embeddings for unlabeled data because encoder part compresses data and decoder part ensures the data can be reconstructed from compressed version again when the whole system is fed with same data as input and output. The network is trained with a function of reconstruction error as loss. Reconstruction error is error between input and reconstructed output from decoder.

Variational Auto-encoders (VAE) [18], differently from aforementioned definition, assure regularized latent space and this provides decoder generative capability. The encoder part of a variational autoencoder fits the input data to a normal distributed latent space and outputs distribution parameters. This makes decoder generate a meaningful output from any random latent variable abiding the constraints.

Teng et al. [8] presented a hybrid system utilizing variational recurrent autoencoder as generator and a rule-based block for temporal production grammar rules. The recurrent VAE is trained with a pop music MIDI dataset including 10000 midi files. The framework has two main capabilities. VAE generates the melody for generated chords by grammar, the rule based part. Secondly, VAE can generate new chord progression for a given melody.

Another VAE-utilizing study is MIDI-VAE by Brunner et al. [9]. MIDI-VAE is a recurrent VAE-based system for style transfer. The system proposed is different than a basic VAE network as the encoder and decoder parts are seperated by a layer. In this layer a style classifier is augmented to latent space, so that encoder learns a compact

"style" label embedded in the latent space.

A difficulty in use of VAE for music generation is long-term sequence modelling. Roberts et al. [19] proposed "hierachical decoder", "which first outputs embeddings for subsequences of the input and then uses these embeddings to generate each subsequence independently". They use bidirectional-LSTM as encoder. In the decoder part, latent space is passed through a "conductor" step which is a fully connected layer with tanh activation and then a conductor RNN which is a unidirectional LSTM. This conductor RNN outputs an embedding vector whose each element is for a subsequence. Finally, the decoder RNN produces each subsequence. This system is called MusicVAE. An important progress in use of AI in art like music is democratization. MidiMe by Dinculescu et al. [20] is an application that utilizes MusicVAE and allows users to generate their own music with updloading their own MIDI files.

3.1.3. Generative Adversarial Networks

Generative Adverserial Networks (GAN), by Goodfellow et al. [21], is a dual-model framework consist of a generator and a discriminator. The generator starts with generating a random noise in data space. The discriminator is trained to differentiate real data from outputs of the generator, while the generator is trained to trick the discriminator. The value function of the whole system training is formulated in the paper as,

$$\min_G \max_D V(D,G) = \min_G \max_D \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[1 - \log D(G(z))] \quad (3.1)$$

Lee et al. [12] proposed a new structure that they focused on generator model, BigVGAN. This study claimed that it provides larger scale than all other studies in the literature with 112M parameters. The generator which is main novelty of this study is composed of multiple convolutional blocks followed by the proposed "anti-aliased multi-periodicity composition (AMP) module". This module mainly applies snake function [22] for periodic inductive bias which significantly improves audio quality. This study shows successful results in voice cloning, voice conversion, speech translation and

audio codec.

Dong et al. [23] presented MuseGAN, multi-track sequential GAN for symbolic music generation. They emphasized the lack of complexity in terms of harmonic and rhythmic structure, multi-track interdependency in the literature. A specific objective metric is used for this interdependency between multiple tracks. Tonal distance, measuring the hamornicity between a pair of tracks. Larger tonal distance implies weaker inter-track harmonic relations.

3.2. Makam Music Related Studies

Due to lack of concensus about symbolic representation or pitch intonations in the Makam music, this type of studies are difficult to gather in the same basis of theory. Parlak et al. [24] developed a makam music generator which uses hardware generated true random numbers. The user chooses the makam, rhythm and instruments of desired music. Random numbers generated are transformed into notes according to makam repository. After the note determining step, a post process is applied for both rhythm and pitch finetuning for melody. Finally, with using Mus2, microtonal notation software, notes are converted to instrument sounds and processed to audio data.

In another study, Parlak et al. [25] proposed a deep learning-based makam composer architecture. This architecture involves three models. Core model is a multi-layer LSTM-fully connected layer model learning to generate sequence as predicting the next pitch-duration tuple with the given sequence. The second model is specialist model which is in the same structure with the core model. However, it is specialized for a certain form, usul and section. After the CM is trained on SymbTR dataset [26] with ignoring these properties, specialist model is trained with transferred initial weights from core model and a smaller dataset prepared focused on these properties. The third model is conductor model which is itself a multi-layer LSTM network. It decides the actual note-duration pair by getting the previous sequence and outputs of two special models. All these LSTM-based networks gets last eight pitch-duration pairs as inputs

to predict the next element. When characteristics of sections in a Makam music performance or composition is considered, this approach specialized according to sections is an attractive feauture in this study. On the other hand, modulations between makams, namely geçki, seems to be ignored or at least can be performed insuffuciently.

3.3. Transformers and Music Generation

In last years, many natural language process (NLP) studies were based on Transformer structure using self-attention mechanism [27]. This architecture completely ditches the recurrent mechanism and is able to use the attention within the encoder also known as the self-attention mechanism. Since it does not utilize a recurrence mechanism to model the sequence information, the model can now process the input tokens in parallel speeding up the training and inference time. But this comes with a cost. Since it does not have the recurrence mechanism, it also does not have a way to encode positional information i.e., hidden representations of the tokens in the sequence $\mathbf{seq_1} = [s_1, s_2, s_3]$ will be the same for the following sequence $\mathbf{seq_2} = [s_3, s_2, s_1]$. In order to alleviate this problem, the original Transformer architecture uses a positional encoding mechanism.

This positional embedding system focuses the order and encodes the absolute positions for tokens. This is not the best scenario for all types of data or problems if we need to deal with relative positions instead of absolute positions. Shaw et al. [28] proposed relative embedding where each token have relative positional embeddings for all other tokens in the sequence, and applied it into self attention. This study shows improvement on machine translation task. However this technique provides advantage in graph-like data representations which is defined with relation between each node pair instead of absolute positions.

We can define the melody generation problem as an NLP task, melodies resemble sentences and pitches are counted as tokens. Despite this analogy, as a long sentence of repetitive pitches, melodies differ sentences that we use in language. With this nature, relative attention sounds more promising comparatively with classical self-attention mechanism. Huang et al. [1] reached successful use of Transformers with relative attention in symbolic music generation.

Radford et al. propose a new language model that utilizes the Transformer decoder and can be pre-trained on a large text corpus called GPT (Generative Pre-Trained Transformer) [29, 30]. In our study, we will work to bring these two concept together: long-term structure with use of relative attention transformer for melody generation and generative pretrained model for different tasks on music.

3.4. Overall Comment on Literature and Contributions

All relative studies in the literature starts with an accepted music theory. In our study, there are no defined tokens for pitches. Our tokens correspond to differentiable frequencies by average human ear. With this aspect, we are trying to find the answer of the question: Can anyone learn to play on a makam with only listening, without theoretical information or constraints given or even musical information about basic notes?

Most studies we mentioned above are MIDI-based works while some of them are end-to-end audio generations or transformations. Although we focus melody generation, this study learns from the performance, not only the simple melody itself. This is a key point for Makam music. We do not focus on synthesize a realistic music, our generated pieces are a sequence of simple sinusoids. However, next steps of this work will be maturing the melody representation with preserving its theory-dependent feature and developing realistic audio synthesizing with Turkish music instruments and compatible with this representation.

4. BACKGROUND

4.1. Self-Attention Mechanism and Transformers

As described in the Transformer paper [27], attention is a mapping function of a query and a set of key-value pairs, where query, keys, values and the output of this mapping are all in the vector space. From high level perspective, this mapping shows which parts of the context should be focused. They proposed scaled dot-product attention which is a modification of commonly used dot-product attention. This new attention is basically computing dot product of query and keys, dividing it by scaling factor which is square root of dimension of the keys, and passing the result through softmax function. The result is weights of values. This mapping is formulated by:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{4.1}$$

where Q, K and V are vectors of query, key and value vectors; and d_k is dimension of key vector k or query vector q.

There is one more modification, which makes the transformers more powerful as attention computations can be done in parallel, which is multi-head attention. There are Q, K and V and attention vector calculated for each token in the sequence in the scaled dot-product attention. However, in multi-head attention, there are multiple linearly projected weight matrices for Q, K and V at the number of heads and resulting multiple attention vectors. These attention vectors concatenated and re-projected into one vector for each token again.

That first transformer proposed in the same paper by Vaswani etl al. [27] is an encoder-decoder structure which relies on the multi-head attention blocks. Both encoder and decoder involves six identical layers.

An encoder layer has a multi-head attention with eight heads and a feed-forward network connected to each other. Each one have residual connection and layer nor-

malization. Attention vectors pass through the feed forward network and transformed into encoded vectors for every word.

Differentially from an encoder layer, a decoder layer has an added block before multi-head attention and feed forward blocks, which is masked multi-head attention. Decoder block continues with a feed forward network ended with softmax resulting with output probabilities of next token.

Residual operation is adding input of the block to the output, mostly to cope with vanishing gradient problem. All layer dimensions are same and 512 to allow residual connections. Additionally, layer normalization after each block is to ensure stability of the weights.

4.2. Relative-Attention and Music Transformer

Shaw et al. [28] improves self-attention mechanism such that it can be utilized for any graph. It is namely, using relative position representation instead of absolute position encoding, and using relation-aware self-attention considering pairwise relation of input elements.

This relation-awareness of self-attention is provided by considering the input as a directed fully-connected graph. It is adding a relative distance element to the self-attention computation in Equation (4.1), S^{rel} which is dot product of Q and R. R is the tensor of embeddings for the relative distances between all keys and queries, $S^{rel} = QR^T$.

Music Transformer [1], first study that is first successfully utilization of transformers in music domain, uses relative attention with memory-efficiency improvements. As implementation details, they used four and eight heads, fixed query, key and value hidden size as in Vaswani et al. [27]. In the experiment, they tuned the number of layers in 4,5,6, attention hidden size in 256, 512 and pointwise feed-forward hidden size in

512, 1024. In the shared implementation, the hyperparameters are selected from them as following: head size: 4, number of layers : 6, sequence length : 2048, embedding dim : 256 (dimensions per head = 256 / 4 = 64).

5. PROPOSED METHOD

Because we use raw audio data but we focus on only melody generation of the transformer, we preprocess the raw data and generate melodies as token sequences. Then using this melody sequences in the training of the transformer and finetune transformers with selected subset of the data for Taksim form in three specific makams. Generation step creates a new sequence for selected time duration and makam, or additionally a given starting sequence.

The whole system can be divided into three main parts, Data Processing, Training, and Generating. Implementation details of these blocks are reported in the next sections and their relations can be seen in the Figure 5.1.

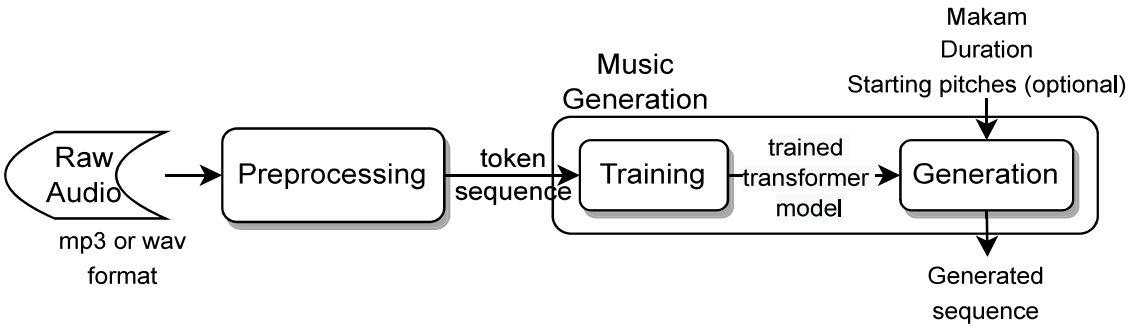


Figure 5.1. Overall system diagram

5.1. Dataset

The dataset is mostly gramophone records from early 1900s resembling the classical genre of this music. there are about 4400 classical performance. About 500 of them are taksims. Dataset contains 159 rast, 75 nihavend and 216 hicaz family records from selected makams. It contains performances with several instruments, orchestras, chorus, and vocal solos.

5.2. Preprocessing

Data preprocessing is all process between the raw audio file and token sequences which are ready to use in learning. It is comprised of resampling, filtering, true-silence detection, melody extraction, tokenization, false silence removal and octave error removal steps.

The main step of data preprocessing is melody extraction. However, some pre and post operations should be added to diminish errors and weakness of the algorithm for our dataset. Additionally, the results should be transformed into the token sequences to proceed in learning.

Our dataset is that we use several gramophone record from early 1900s to 1950s. These records are generally very noisy and need filtering. A filtering is applied before melody extraction. Even so, some data could not be transformed into melodies because of this noise.

Another issue is estimating karar frequency. Unlike the tonal music, makam music does not have a "tone" which can be estimated statistically from the end of the record. However, each piece or performance finish with the karar tone of the makam. Half of this problem is again melody extraction problem because voicing errors or octave errors in the end of performance result with wrong estimation of karar frequency. Estimating the karar is important because, singers generally perform from different transpositions. Instrumental performances can also be transposed from the original frequencies. Additionally, there are some total frequency changes depending on recording. Thus, each record should be transformed into frequency ratios according to karar pitch and then tokenized according to ratios.

One of the main problems in melody generation is voicing errors. Used framework for melody extraction labels the frames it could not find the true frequency as silence. This causes the original silences of the record and the silences coming from voicing

errors are mixed after the melody extraction step. To avoid that, we detect the silences of the record after filtering step and use the true silences to remove false silences emerged in the melody extraction after tokenization step.

Another main problem of melody generation is octave errors. Especially Turkish traditional instruments and classical singing technique used by the singer in the low frequencies results in the second harmonic (octave of fundamental freq.) often having a larger amplitude than the fundamental frequency and this cause octaves errors with correct pitch occur frequently [31]. Even if the melody extraction framework involves octave error removal part itself, it is observed that it is not enough for our domain and data. Thus, an octave error removal step based on tokens takes part in preprocessing.

The whole preprocessing is depicted with the system diagram in 5.2. Each block on the diagram will be detailed in the following sections.

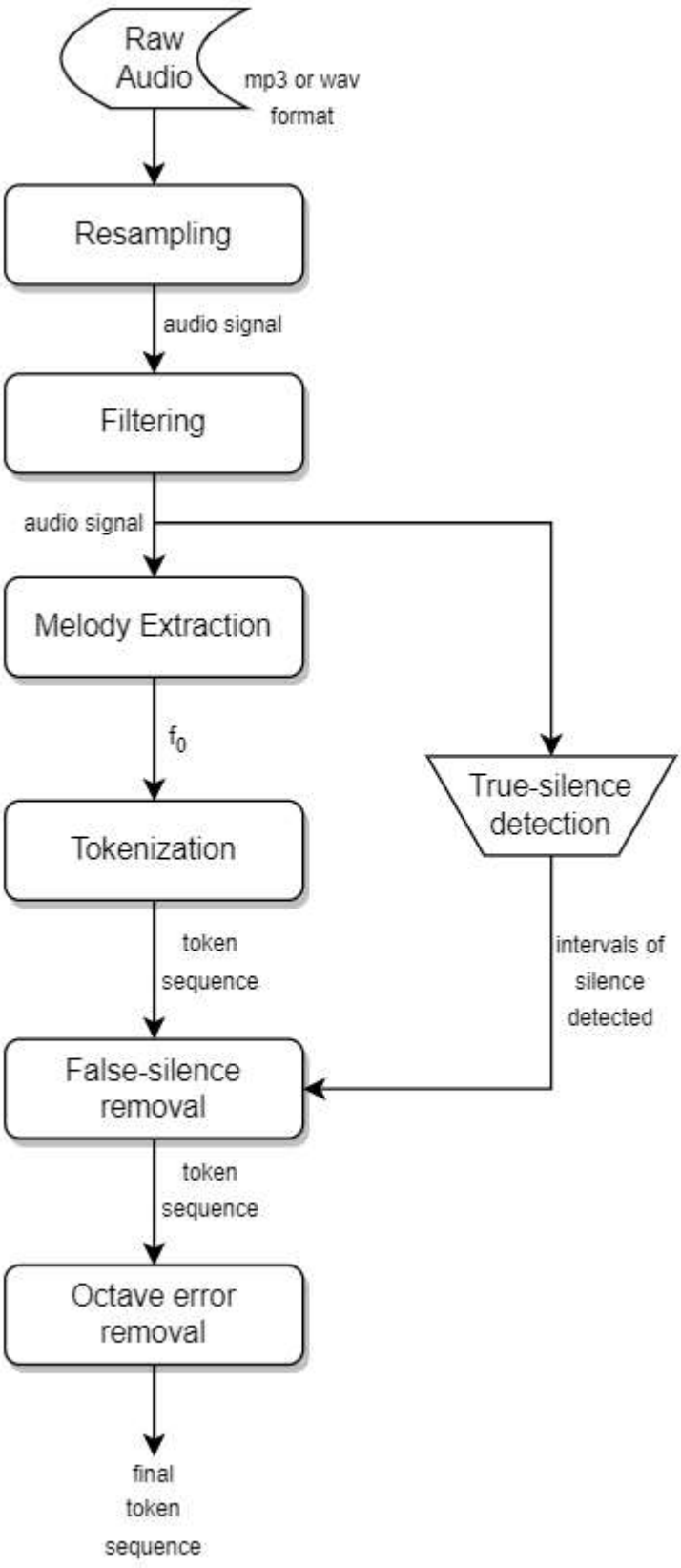


Figure 5.2. System diagram of preprocessing

5.2.1. Resampling

Sampling rate is the only decision to take in resampling step. In the algorithm used in melody extraction, the supported sampling rates are 48kHz, 44.1kHz, 32kHz, 24kHz, 22050Hz, 16kHz, 12kHz, 11025Hz, 8kHz and 44.1kHz. is recommended. After different trials, 16kHz is selected because there are not a drastically reducing performances with decreasing sampling rates and 44.1kHz increase time and memory need.

5.2.2. Filtering

The filtering step has two goal, diminishing the noise of gramophone records and helping the melody extraction algorithm for less octave errors. Noise of gramophone records is in low frequencies and we need to remove them. To diminishing octave errors we need to attenuate the higher frequencies. A bandpass filter with different orders in high and low edge is constructed with cascading a high-ordered Butterworth lowpass filter and low-ordered Butterworth highpass filter. We choose the orders eight and two, cutoffs 60 and 220 respectively.

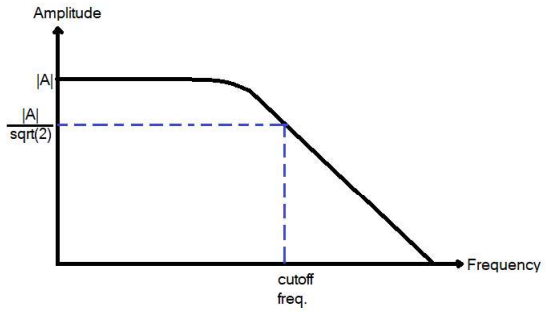


Figure 5.3. Frequency response of Lowpass Butterworth filter

Butterworth lowpass filter is a non-ideal filter having a lower response for higher frequencies along the whole frequency range as depicted in Figure 5.3. This response diminishes the powerful $2f_0$'s more than f_0's and reduce octave errors.

5.2.3. True-silence Detection

Root mean square of the resampled and filtered audio signal is calculated with window size 1000, and hop length 200. Less than 20 dB below of maximum is considered as silence. Silence intervals are the output of this stage and used in false-silence removal.

5.2.4. Melody Extraction

Melody extraction is the task of estimating the fundamental frequency corresponding to the pitch of the main melodic line in a music. So, melody extraction is the work that digital audio signal is processed and results with a sequence of pitch frequencies.

There are various melody extraction algorithms in the literature. Some popular and easy to apply ones are YIN (not an abbreviation, coming from yin and yang), pYIN, MELODIA, variational mode decomposition (VMD). In the study presents a modified version of VMD for Turkish makam music focused melody extraction, the results of mentioned algorithms are compared [32]. According to some trials on these algorithms and this comparison, MELODIA [33] is selected. The main reasons are voicing success and pitch accuracy when octave errors are ignored.

The mentioned architectures focus on different dynamics or characteristics. As an example, while YIN [34] is based on autocorraletion and post error removal steps, pYIN [35] outputs several probabilistic YIN results and uses them for an improved pitch pitch sequence with using a hidden Markov Model.

MELODIA algorithm focuses the pitch contour characteristics. Pitch contours are time continuous sequences of pitch candidates. A set of contour characteristics are defined and used to determine some rules to distinguish between melodic and non-melodic contours. This leads to the development of new voicing detection, octave error minimization and melody selection techniques [33].

Proposed method consists of four main stages. They are sinusoid extraction, salience function, contour creation and melody selection.

The Sinusoid Extraction stage primarily employs the Fourier Transform to identify the harmonics that constitute a musical sound. Prior to the application of the transform, an "Equal Loudness Filter" is utilized to enhance frequencies that are more easily perceived by the human ear. The filter is constructed by cascading a tenth-order Infinite Impulse Response filter with a second-order Butterworth High-Pass filter. Subsequently, the Short-Time Fourier Transform (STFT) is applied, which involves the iterative use of the Fast Fourier Transform (FFT) in a moving window with a windowing function. However, the results of the STFT are often inaccurate as the spectral peak locations are limited to the bin frequencies of the FFT. To address this issue, the final step of the Sinusoid Extraction stage is the Frequency/Amplitude Correction of the STFT results. The phase spectrum is employed to calculate the peak's Instantaneous Frequency (IF) and amplitude, which provide a more precise estimate of the peak's true frequency and amplitude, as outlined by Dressler [36].

The results of first stage, frequencies and amplitudes of spectral peaks are used for a representation of pitch salience over time in the second stage. Harmonics are generally construct a musical sound together. This stage mainly computes the salience for each frame as the sum of the weighted energies found at harmonics.

After the salience function stage, third one, pitch contour creation is executed. Pitch contour is a sequencing of pitch values which are continuous in both time and frequency. The peaks of the salience function are taken at each moment in time with peak filtering and streaming steps and transformed into pitch contours.

The final stage in the process is the Melody Selection stage. This stage encompasses several tasks, including Voicing Detection, Melody Peak Selection, and iterative Octave Error Outlier Removal. The latter is used to calculate a "Melody Pitch Mean", which represents the large-scale temporal evolution of the melody. Furthermore, the

peaks that belong to the main melody are selected. In most cases, there is only a single peak to choose from. However, in instances where more than one contour is present within a frame, the melody is chosen as the peak belonging to the contour with the highest overall salience. If no contour is present in the frame, it is considered to be silent.

5.2.5. Tokenization

After obtaining the f_0 sequences, frequency of the last pitch played in each record, karar frequency, is detected. Each record is transformed to a list of ratios of each frame f_0 according to karar frequency.

The term token is generally used as words in vocabulary in Natural language processing. If we make an analogy and assume melody generation as an NLP task, token resembles pitch.

As mentioned in the theory section, there are different theories about microtone locations and also a definition difficulties of each microtone due to makam and seyir differences. To make this study theory-independent in microtone location aspect, we define the tokens according to average human ear capability to distinguish the pitches and assign as many tokens as possible for microtone candidates. In logarithmic scale of frequencies, one octave is assumed as 1200 cent by mathematical definition. Average human distinguish capacity of two sound is about 4-5 cent. On the other hand, main tones are defined with Pythagorean scale which a minor second interval (semitone) is 90 cent, a major second interval is 204 cent approximately. Thus we define tokens such as each interval between two neighbor tokens is four or five cents. In one octave, there are 255 tokens. The vocabulary includes 1023 tokens. Token-0 represents silence, token-1022 represents the decision pitch, and the others represent two octaves above and below the decision pitch. Each frequency rate obtained in the record is quantized according to the rate values of the token dictionary and converted into the corresponding token.

To clarify the tokenization idea in this study, we can make a comparison. If we study the classical Western music or any genre using 12-tone equal temperament system, we would have 12 tokens for each octave. However, we should have had a system that learn and generate polyphony and harmony. On the other hand, for Makam music, we allow the system to learn and generate melody with all possible microtone candidates depending to makams.

5.2.6. False Silence Removal

As mentioned in the previous stages, melody extraction step results with some false silence decision and several erroneous octave jumps. While false silence errors reduce the melodic continuity, octave errors directly changes the melody. False silences should be removed both to obtain more continuous melody and to make the octave error removal step more efficient.

False silences are result of failures of voicing detection stage in the melody extraction part. With true-silence detection step after filtering, our mechanism knows actual silence coming from record. False silence error is removed basically with the last pitch before silence.

5.2.7. Octave Error Removal

Octave errors generally result with the second harmonic ($2f_0$) instead of the first harmonic (f_0). It is observed that as Butterworth filter gets closer to ideal low pass filter with higher degrees, the accuracy of MELODIA in the other regions has declined. Thus, we decide to use the filter but also add another stage because the filter only remove few of octave errors.

The main octave error removal takes part after tokenization. In this stage a melody-centric algorithm act as following: A window size is given as input to algorithm. Let token list for a record be named T and there be a jump about one octave between

ith and $i+1$th element of T. Algorithm takes the window afterT_i and decide if T_{i+1} is octave error or not.

As an example result, a taksim before and after removals is depicted in the Figure 5.4. False silences and most octave errors are removed successfully. A highly stable octave error which could not be removed because of length is shown in the after removal melody chart.

5.2.7.1. Quantitative Analysis of Octave Error Removal. To report a quantitative analysis on success of octave error removal, we need ground truth of data which is too difficult for whole dataset. A sample dataset is selected and the quantitative analysis is done on this dataset. To ensure diversity, examples from different makams, forms and instrumental-vocal combinations are selected. four taksims with different instruments and makams(ussak kemence taksimi, hicaz tanbur taksimi, hüzzam kanun taksimi, rast ney taksimi), two orchestral instrumental records and two vocal (one from male singer, one from female singer) records are selected. Pre-removal and post-removal octave error ratios in terms of sample counts are reported on the Table 5.1.

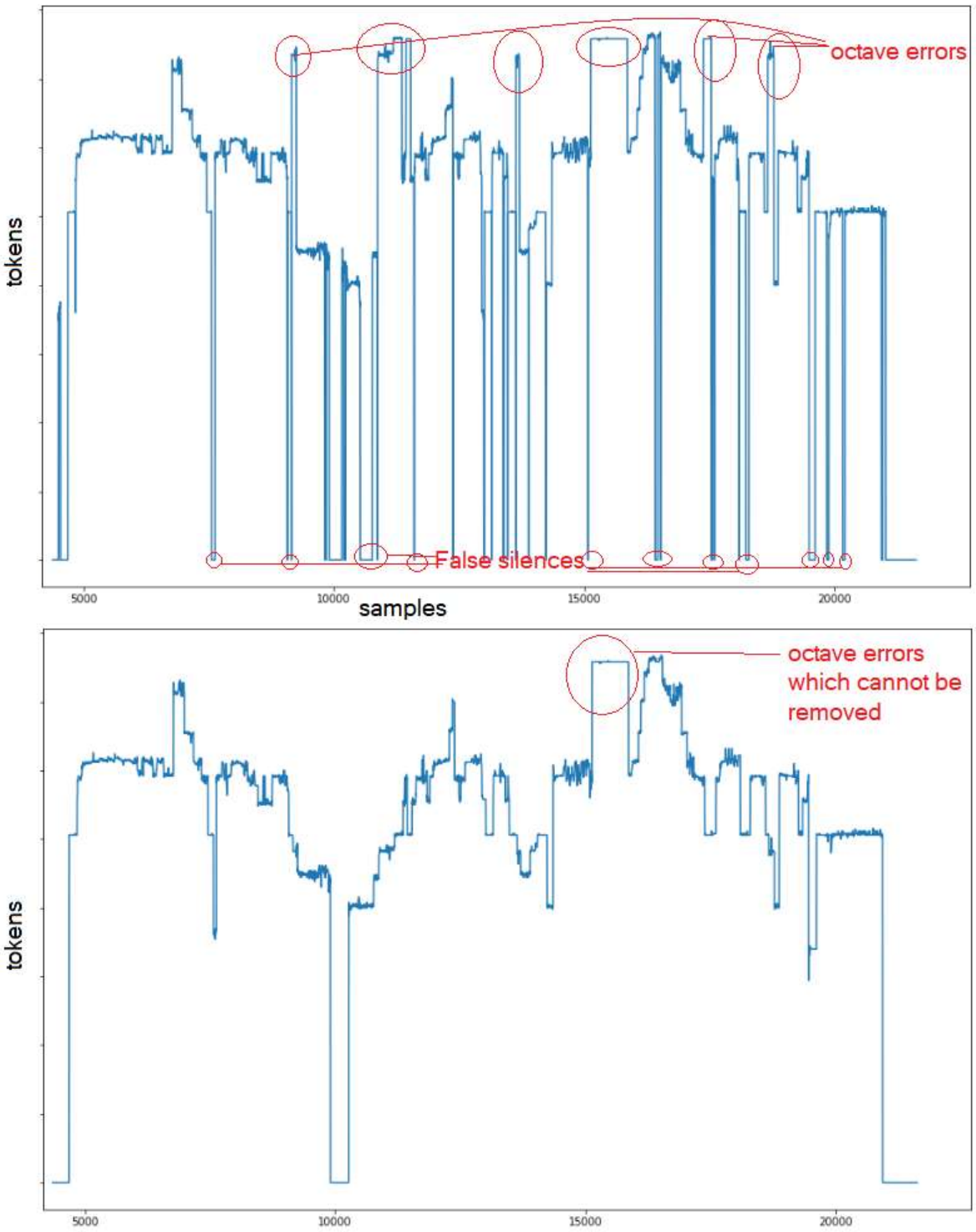


Figure 5.4. Before (above) and after (below) octave error and false silence removal example on a Taksim

As seen in the Table 5.1, the removal success of the octave error removal block is 77% in average. The ratio of octave errors which cannot be removed in whole systems is 4%

Table 5.1. Octave error removal statistics on the sample set.

Record	Sample Count	Before Removal	After Removal	Removal Success
Uşşak taksim (kemence)	30237	3055 (10%)	1218 (4%)	60%
Rast taksim (ney)	80440	12428 (15%)	4941 (6%)	60%
Hicaz taksim (tanbur)	70588	7500 (11%)	1635 (2%)	78%
Hüzzam taksim (kanun)	17141	323 (2%)	139 (0.8%)	57%
Rast Peşrev (orch.)	80440	24890 (31%)	3828 (5%)	85%
Muhayyer Saz Semaisi (orch.)	91443	22500 (25%)	5992 (7%)	73%
Hisar Murabba (female vocal)	111820	20964 (19%)	4250 (4%)	80%
Suzinak Şarkı (male vocal)	86773	2172 (3%)	870 (1%)	60%
Overall		15%	4%	77%

5.3. Training

The data as in the form of sequences of tokens is used to train a Transformer with relative attention as proposed by Huang et al. [1] and detailed in the Section 4.2. The significant difference of our training step from this study is data representation. As detailed in the section 5.2.5, we tokenize pitches in the melody as all possible microtones, and to cover nuances in the performance we do not transform the pitch sequences into sequences of pitch-interval pairs. Rather than that, we directly feed the sequences into the model. Vocabulary size is 1023 in total, where 0 is silence, 1022 is end-of-record token and 1-1021 are the microtones derived as in the Section 5.2.5.

We train the Transformer with eight heads, 512 embedding dimensions, eight layers and sequence window length is 3480 which is 10 seconds of data. The hyperpa-

rameters selected intuitively by comparing the our data representation and needs with the Music Transformer study, also taking into account the possibilities of GPU use. Training is done in Google Colab server with premium GPU's.

We train the network with two main steps, one is general learning with the whole data and randomly chosen sequences. The second one is makam-specific learning with only records in selected makams. The general learning is run for 3.7m learning steps where each step is with one sequence in given sequence length. Same model continues to learn in three separate learning paths 500k steps more and results with three different makam models.

5.4. Generation

Generation module runs according to the makam, selection of the model to use, a duration which actually defines maximum sequence length, and a starting sequence.

Maximum sequence length is assigned as such the time duration of the melody is one minute. This melody duration is decided for some reasons. First, it is feasible and matches our assumptions and limitations. Because we do not separate the training data sequences into parts according to modulations, our models are not aware of the modulation concept, instead generate similar patterns in the main scale of the selected makam. With the same data utilization, converting the model to a modulation-aware system needs highly larger model and longer training. Thus, we aimed half to one minute-length taksims with few or no modulations. Additionally, this duration also shows parallelism with similar studies in the literature, which our base study by Huang et al. [1] is included.

Starting sequence can is given into the model to use windowing sequence at the beginning but it can be all-zeros sequence which is silence. Thus it is an optional parameter in our system.

Given sequence can be in any length, first it is padded with zeros for a sequence length of 10 seconds, which is training sequence window length. In each iteration, the model generate next element according to this window sequence and the sequence is updated. There are two termination criteria which results with we can have different length than the maximum sequence length. If the generator predicts end of sequence token, it makes generation stop. The other criteria is the model is stationary at the karar token which is same with the makam characteristics. These criteria results with more plausible melodies for taksim form.

Synthesizing audio signals from generated sequence is basically generating sinusioids from the frequencies. Because our study focuses on generating melody symbolically rather than generating realistic sounds, we use a simple synthesizer that transforms sequence of pitch frequencies to sequence of sinusoids. The result is monophonic audio signal that directly consists of fundamental harmonics of pitch frequencies. This make the whole system easy to output results in development. On the other hand, it effect evaluation results because people expect more realistic sounds and evaluate the results not only for melodies but also this weird sound.

The generation module operates according to the makam, the selection of the model to be used, a duration which determines the maximum sequence length, and a starting sequence. The maximum sequence length is set such that the duration of the melody is one minute. This duration was chosen for several reasons. Firstly, it is feasible and aligns with our assumptions and limitations. Since we do not segment the training data sequences according to modulations, our models are not aware of the concept of modulation and instead generate similar patterns in the main scale of the selected makam. Converting the model to a modulation-aware system with the same data usage would require a much larger model and longer training time. Therefore, we aimed to generate half to one minute-length taksims with few or no modulations. In addition, this duration also reflects parallelism with similar studies in the literature, including our base study by Huang et al. [1].

The starting sequence can be provided to the model as a windowing sequence at the beginning, but it can also be an all-zero sequence, representing silence. Thus, it is an optional parameter in our system. The given sequence can be of any length; it is first padded with zeros to a sequence length of 10 seconds, which is the training sequence window length. In each iteration, the model generates the next element based on this window sequence, and the sequence is updated. There are two termination criteria that can result in a sequence length different from the maximum sequence length. If the generator predicts the end of sequence token, it stops the generation. The other criteria is the model is stationary at the karar token, which is characteristic of the makam. These criteria result in more plausible melodies in the taksim form.

Synthesizing audio signals from the generated sequence involves generating sinusoids from the frequencies. As our study focuses on generating melodies symbolically rather than generating realistic sounds, we use a simple synthesizer that transforms a sequence of pitch frequencies into a sequence of sinusoids. The result is a monophonic audio signal that consists directly of the fundamental harmonics of pitch frequencies. This makes the entire system easy to output results during development. However, it affects the evaluation results because people expect more realistic sounds and evaluate the results not only based on the melodies, but also on the sound quality.

6. EXPERIMENT AND RESULTS

The experiment is based on the generation results of model in three makams, Rast, Nihavend and Hicaz in half to one minute-length. These three makams are selected because they are the makams having largest examples in the dataset. Additionally, these makams are simple makams with a characteristics scale if we do not consider modulations. As discussed before, we utilize the pieces as a whole, not as fragments. Because of that, this study gives limited results in terms of modulation ability between makams. Additionally, the main characteristics of these simple makams ease the subjective evaluation of the musicians attending the evaluation survey. Lastly, when we compare the experiment setup in terms of makam number used in generation with the similar studies in literature, it is quite satisfactory. The makam related studies restricted the makams with a few of them, as an example, Parlak et al. [24] selected two makams for simplicity of experiment.

The half to one-minute duration is a good starting point as a simple taksim with a few or no modulations. Five taksims per each makam, total 15 taksims are cherry-picked among generation trials as experiment set. Our evaluation procedure consists of two steps which are subjective evaluation which results are collected by 10 volunteer amateur-professional Turkish Makam musicians.

It should be realized that using monophonic synthesizer made the human evaluation difficult and their results lower. Also the study was not completely explained to volunteers. They were only informed about the focus target of this study, synthesizer issue mentioned above and gave an example for that with a real taksim as original version and its synthesizer output as a comparison. Besides, the volunteers do not have exactly same background, theoretical basis and focuses for a taksim performance. To overcome those points and make a more analytic evaluation we added some some pieces from real taksim performances by Makam music virtuosos into experiment set as control data. The numerical analysis of subjective evaluations is done by normalizing

each volunteer votes according to their votes for control group.

The first evaluation is 1-10 voting for each sample as similar some example in the literature and identical to Huang et al. [17]. 1 is noise, 5 is plausible music and 10 is "improvised by a real musician". This part does not focus taksim form and makams, it asks only votes for the melody as an musical improvisation. Each vote of volunteer is standardized as the control group is voted as 10. Overall melody rating is 6.88 in average with 3.16 standard deviation as the points of results are changing between about 3 and 9.5.

The second one is makam-related human evaluation for each record. Attendees are asked to make a choice for makam of the listened taksim. If the record does not sound like a taksim in given makams they vote as "None of them". In the continuation, if they select a makam, they are asked to vote 1-10 point for how the taksim compatible is according to pitches and how it is compatible according to melodic progression, with the selected makam. This evaluation results with three metrics. Makam accuracy, pitch compatibility and melodic progression compatibility. Makam accuracy is true positive ratio of the makam selection for each makam separately and their average as a whole accuracy. Compatibility metrics are again 1-10 votes standardized according to control group. These are also calculated separately for each makam and as overall. These results can be seen in the Table 6.1.

Table 6.1. Human evaluation Results

Model Results	Melody Rating	Makam Accuracy (0-1)	Pitch Compatibility	Melodic Progression Compatibility
Hicaz	7.54 ± 1.23	0.70 ± 0.17	6.00 ± 0.96	5.35 ± 0.97
Nihavend	6.31 ± 1.53	0.60 ± 0.17	6.99 ± 1.40	5.32 ± 0.78
Rast	6.79 ± 1.98	0.73 ± 0.12	7.20 ± 1.35	6.70 ± 1.55
Overall	6.88 ± 1.54	0.68 ± 0.13	6.73 ± 1.22	5.79 ± 1.20

We find that the model generating taksim in hicaz makam performs better in melody ratings without considering makam fitness. This may be due to the large amount of hicaz data. However, it performs worse than the other two models in terms of makam fitness of pitch and melodic progression, despite having similar accuracy to Rast and better accuracy than Nihavend. This may be because the distinct pitch ranges in the hicaz create more complex patterns that are difficult for the model to summarize.

Despite the better performance of the hicaz model in melody ratings, the model generating rast shows better results in other comparisons. The nihavend model has the worst results in terms of melody rating, makam accuracy, and melodic progression compatibility with the makam. This may be due to the scarcity of Nihavend records in the dataset compared to other makam records.

Overall, the taksim generator produces plausible melodies and learns three main makams at a satisfactory level. We had originally planned to conduct a numeric evaluation of makam-pitch compatibility as an objective evaluation, but this approach was not suitable due to the imitative nature of the model. Instead, we added compatibility questions to the survey and standardized the results, which were acceptable. The generation examples can be reached by the evaluation survey [37].

7. CONCLUSION & FUTURE WORK

Microtonal music, specifically Makam music, is a fascinating subject for melody generation tasks. In this study, we investigate the use of artificial intelligence in this genre to learn the theoretical background of Makams and gain improvisational skills through variations. We also explore a theory-independent learning approach using the Transformer architecture in melody generation for Makam music.

This study presents a transformer-based approach to the generation of makam music melodies. By training a transformer model on preprocessed and melody-extracted data from real performances, and finetuning the model for the generation of Taksim, a musical form related to improvisation in specific makams, the proposed method is able to generate makam melodies that are faithful to the traditional style and structure of the music. This work represents a significant advancement in the application of deep learning techniques to the generation of non-Western musical styles, and has the potential to inspire further research in this area.

As a criticism of this study, the preprocessing is not entirely successful due to the focus on the concept of transformer use in microtonal music. The use of gramophone records with high levels of noise requires more sophisticated preprocessing to enable the extraction of melodies with a more error-free system. More makam Music-oriented and noise-robust melody extraction methods should be developed as future work in this area. Additionally, our octave error removal and false silence removal steps are totally differentiable from the pipeline. They can be transformed into more generalized functionalities and improved by deep learning base methods. Additionally, our learning system is not capable of generating complex compositions with appropriate modulations according to the makam. To achieve this, the data must be more detailed, the modulations must be labeled with makams, and the method must be more modulation-aware. However, we consider the whole melody of each record to be in the main makam.

The next step of this study is the development of a scalable microtonal music environment for theory-independent and performance-oriented studies, as the concept is proven here. We plan to construct this scalable system and improve the preprocessing in this work, enabling different research areas around Makam music.

REFERENCES

1. Huang, C.-Z. A., A. Vaswani, J. Uszkoreit, N. Shazeer, I. Simon, C. Hawthorne, A. M. Dai, M. D. Hoffman, M. Dinculescu and D. Eck, "Music Transformer", *arXiv preprint arXiv:1809.04281*, 2018.

2. Liang, F., "BachBot: Automatic Cmposition in the Style of Bach Chorales", *University of Cambridge*, Vol. 8, pp. 19–48, 2016.

3. Eck, D. and J. Schmidhuber, "Finding Temporal Structure in Music: Blues Improvisation with LSTM Recurrent Networks", *Proceedings of the 12th IEEE Workshop on Neural Networks for Signal Processing*, pp. 747–756, IEEE, 2002.

4. Huang, Y.-S. and Y.-H. Yang, "Pop music transformer: Beat-based Modeling and Generation of Expressive Pop Piano Compositions", *Proceedings of the 28th ACM International Conference on Multimedia*, pp. 1180–1188, 2020.

5. Wang, J., X. Wang and J. Cai, "Jazz Music Generation Based on Grammar and LSTM", *2019 11th International Conference on Intelligent Human-Machine Systems and Cybernetics*, Vol. 1, pp. 115–120, IEEE, 2019.

6. Eck, D. and J. Schmidhuber, "A First Look at Music Composition using LSTM Recurrent Neural Networks", *Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale*, Vol. 103, p. 48, 2002.

7. Kalingeri, V. and S. Grandhe, "Music Generation with Deep Learning", *arXiv preprint arXiv:1612.04928*, 2016.

8. Teng, Y., A. Zhao and C. Goudeseune, "Generating Nontrivial Melodies for Music as a Service", *arXiv preprint arXiv:1710.02280*, 2017.

9. Brunner, G., A. Konrad, Y. Wang and R. Wattenhofer, "MIDI-VAE: Modeling Dy-

namics and Instrumentation of Music with Applications to Style Transfer", *arXiv preprint arXiv:1809.07600*, 2018.

10. Grekow, J. and T. Dimitrova-Grekow, "Monophonic Music Generation with a Given Emotion using Conditional Variational Autoencoder", *IEEE Access*, Vol. 9, pp. 129088–129101, 2021.

11. Choi, K., C. Hawthorne, I. Simon, M. Dinculescu and J. Engel, "Encoding Musical Style with Transformer Autoencoders", *International Conference on Machine Learning*, pp. 1899–1908, PMLR, 2020.

12. Lee, S.-g., W. Ping, B. Ginsburg, B. Catanzaro and S. Yoon, "BigVGAN: A Universal Neural Vocoder with Large-Scale Training", *arXiv preprint arXiv:2206.04658*, 2022.

13. Oord, A. v. d., S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior and K. Kavukcuoglu, "Wavenet: A Generative Model for Raw Audio", *arXiv preprint arXiv:1609.03499*, 2016.

14. Briot, J.-P., G. Hadjeres and F.-D. Pachet, "Deep Learning Techniques for Music Generation - a Survey", *arXiv preprint arXiv:1709.01620*, 2017.

15. Isaacson, L. M., *Experimental Music; Composition with an Electronic Computer by Lejaren A. Hiller, Jr. and Leonard M. Isaacson*, New York, McGraw-Hill, 1959.

16. Chen, C.-C. and R. Miikkulainen, "Creating Melodies with Evolving Recurrent Neural Networks", *International Joint Conference on Neural Networks. Proceedings*, Vol. 3, pp. 2241–2246, IEEE, 2001.

17. Huang, A. and R. Wu, "Deep Learning for Music", *arXiv preprint arXiv:1606.04930*, 2016.

18. Kingma, D. P. and M. Welling, "Auto-encoding Variational Bayes", *arXiv preprint*

arXiv:1312.6114, 2013.

19. Roberts, A., J. Engel, C. Raffel, C. Hawthorne and D. Eck, "A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music", *International Conference on Machine Learning*, pp. 4364–4373, PMLR, 2018.

20. Dinculescu, M., J. Engel and A. Roberts, "MidiMe: Personalizing a MusicVAE Model with User Data", *Workshop on Machine Learning for Creativity and Design*, 2019.

21. Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, "Generative Adversarial Networks", *Communications of the ACM*, Vol. 63, No. 11, pp. 139–144, 2020.

22. Ziyin, L., T. Hartwig and M. Ueda, "Neural Networks Fail to Learn Periodic Functions and How to Fix It", *Advances in Neural Information Processing Systems*, Vol. 33, pp. 1583–1594, 2020.

23. Dong, H.-W., W.-Y. Hsiao, L.-C. Yang and Y.-H. Yang, "Musegan: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment", *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32, 2018.

24. Parlak, İ. H. and C. Kösemen, "Automatic Music Generation by True Random Numbers for Turkish Makams", *2018 4th International Conference On Computer And Technology Applications*, pp. 64–68, 2018.

25. Parlak, I. H., Y. Çebi, C. Işıkhan and D. Birant, "Deep learning for Turkish makam music composition", *Turkish Journal of Electrical Engineering and Computer Sciences*, Vol. 29, No. 7, pp. 3107–3118, 2021.

26. Karaosmanoğlu, M. K., "A Turkish Makam Music Symbolic Database for Music Information Retrieval: SymbTr", *Proceedings of 13th International Society for Mu-*

sic Information Retrieval Conference; 2012 October 8-12; Porto, Portugal. Porto: ISMIR, 2012. p. 223–228., International Society for Music Information Retrieval (ISMIR), 2012.

27. Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser and I. Polosukhin, "Attention is All You Need", *Advances in Neural Information Processing Systems*, Vol. 30, 2017.

28. Shaw, P., J. Uszkoreit and A. Vaswani, "Self-attention with Relative Position Representations", *arXiv preprint arXiv:1803.02155*, 2018.

29. Radford, A., K. Narasimhan, T. Salimans, I. Sutskever *et al.*, "Improving Language Understanding by Generative Pre-training", , 2018.

30. Radford, A., J. Wu, R. Child, D. Luan, D. Amodei and I. Sutskever, "Language Models are Unsupervised Multitask Learners", *OpenAI blog*, Vol. 1, No. 8, p. 9, 2019.

31. Salamon, J., E. Gómez, D. P. Ellis and G. Richard, "Melody Extraction from Polyphonic Music Signals: Approaches, Applications, and Challenges", *IEEE Signal Processing Magazine*, Vol. 31, No. 2, pp. 118–134, 2014.

32. Şimşek, B. Ö. and A. Aydın, "Monofonik Türk Makam Müziği Eserlerinde İyileştirilmiş Değişken Kip Ayrışım Yöntemi ile Temel Frekans Kestirimi", *Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi*, Vol. 33, No. 4, pp. 1333–1346, 2018.

33. Salamon, J. and E. Gómez, "Melody Extraction from Polyphonic Music Signals using Pitch Contour Characteristics", *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 20, No. 6, pp. 1759–1770, 2012.

34. De Cheveigné, A. and H. Kawahara, "YIN, a Fundamental Frequency Estimator for Speech and Music", *The Journal of the Acoustical Society of America*, Vol. 111,

No. 4, pp. 1917–1930, 2002.

35. Mauch, M. and S. Dixon, "pYIN: A Fundamental Frequency Estimator using Probabilistic Threshold Distributions", *2014 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 659–663, IEEE, 2014.

36. Dressler, K., "Sinusoidal Extraction using an Efficient Implementation of a Multi-resolution FFT", *Proc. of the Int. Conf. on Digital Audio Effects (DAFx-06)*, pp. 247–252, Citeseer, 2006.

37. Bilgic, T., "Yapay Zeka Ile Taksim üretimi Değerlendirme Anketi", 2023, `https://forms.gle/bpMPPnP1cUg162f36`, accessed on January 18 2023.