





## BSc Thesis Task Description

**Mohammaderfan Koupaei**

candidate for BSc degree in **Computer Engineering**

# Music Generation With Deep Learning Algorithms

Embarking on a melodic exploration of artificial intelligence, this project seeks to create a deep learning model that crafts beautiful and coherent music, inspired by textual cues or initial melody snippets. In simpler terms, imagine writing a line of poetry or humming a few notes and having a computer turn that into a full musical piece! By training the model with a wide variety of songs and melodies from datasets like Lakh MIDI, it will learn to recognize different musical elements and styles, ensuring the music it creates is not only technically accurate but also pleasing to listen to. The aim is not just to explore how well AI can create music, but also to understand its potential role in future music creation and production, providing insights and tools that could be used by musicians, composers, and anyone interested in music generation.

- **Model Development**

Utilize RNNs and LSTMs, leveraging their ability to manage sequential data, to develop a model capable of interpreting and generating music from textual or melodic inputs.

- **Study and Analyze**

Leverage existing studies and utilize tools and datasets like MuseScore3, Musenet, and the Lakh MIDI dataset for training and sampling purposes, ensuring a robust foundation and diverse training ground for the model.

- **Feature Extraction**

Employ MIDI representations to focus on musical notes, durations, and other relevant attributes during model training.

- **Evaluate**

The generated melodies will be assessed for their similarity to human-composed music, targeting an accuracy of 90% or higher, to ensure the AI-created compositions maintain a human-like quality and musical coherence

- **Outcome**

The project aims to provide a nuanced understanding of AI's capabilities in music generation, offering a platform for users to generate music and providing foundational knowledge for future research in AI-driven creative endeavors

**Supervisor at the department:** **László György Grad-Gyenge, assistant research fellow**

**External supervisor:**

Budapest, 17 october 2023

/ Dr. Hassan Charaf /  
professor  
head of department





**Budapest University of Technology and Economics**  
Faculty of Electrical Engineering and Informatics  
Department of Control Engineering and Information Technology

Mohammaderfan koupaei

**MUSIC GENERATION USING  
DEEP LEARNING MODELS**  
SEQ2SEQ TRANSFORMER

SUPERVISOR

**Grad-Gyenge László György**

BUDAPEST, 2023

## Contents

<b>Summary.....</b>	<b>5</b>
نگاهی کوتاه.....	6
<b>1 Introduction.....</b>	<b>7</b>
1.1 Motivation.....	7
1.2 Objectives and questions .....	7
1.3 Thesis structure .....	8
<b>2 Related works and literature review .....</b>	<b>9</b>
2.1 Melody line extraction .....	9
2.2 Lyrics to Melody Generation .....	10
<b>3 Background knowledge .....</b>	<b>12</b>
3.1 Music theory .....	12
3.1.1 MIDI file .....	13
3.1.2 Pitch .....	13
3.1.3 Duration .....	14
3.1.4 Rest .....	14
3.1.5 Tonality .....	15
3.1.6 Chord progression.....	15
3.2 Lyrics and melody .....	16
3.2.1 Syllable and phoneme .....	16
3.2.2 Natural Language Processing .....	17
3.3 Technical overview .....	19
3.3.1 Transformers .....	19
3.3.2 Sequence-to-sequence Transformer model.....	19
<b>4 Methodology and model design .....</b>	<b>21</b>
4.1 Data collection and preprocessing .....	21
4.1.1 Dataset .....	21
4.1.2 Preprocessing .....	21
4.2 Model design.....	29
4.2.1 Model architecture .....	29
<b>5 Training .....</b>	<b>35</b>
5.1 Setup .....	35
5.2 Training Steps .....	35
<b>6 Evaluation and results .....</b>	<b>37</b>

6.1 Quantitative analysis .....	37
6.1.1 Pitch Analysis .....	37
6.1.2 Rest Analysis .....	39
6.1.3 Duration Analysis .....	40
6.1.4 Melody Consistency .....	40
6.2 Qualitive analysis.....	42
<b>7 Conclusion .....</b>	<b>45</b>
<b>References .....</b>	<b>47</b>

The Figure 3.1 .....	18
Figure 3.2 Simple transformer architecture[42] .....	19
Figure 3.3 seq2seq transformer architecture .....	20
Figure 4.1preprocessing pipeline .....	22
Figure 4.2 Midi score with two track.....	23
Figure 4.3 lyrics processing pipe line .....	24
Figure 4.4 sequence of pitches in a midi score .....	25
Figure 4.5 pipe line of melody processing.....	26
Figure 4.6 process of melody and lyrics extraction .....	27
Figure 4.7 word2vec embedding for love .....	27
Figure 4.8 word embedding visualized with t-Sne and PCA.....	28
Figure 4.9 sentence , word , syllable distribution .....	28
Figure 4.10 sentence , word, syllable boxplot distribution .....	29
Figure 4.11 seq2seq transformer model with attention mechanism[49].....	30
Figure 5.1 process of attention mechanism .....	36
Figure 6.1 Distributions of rest durations and intervals.....	38
Figure 6.2 Various comparisons of pitch distributions and progressions.....	38
Figure 6.3 Various comparisons of Rest distributions and progressions.....	39
Figure 6.4 various comparison Duration distributions .....	40
Figure 6.5 pitch repetition in test and generated data .....	41
Figure 6.6 generated melodies .....	44

# STUDENT DECLARATION

I, **Mohammaderfan koupaei**, the undersigned, hereby declare that the present BSc thesis work has been prepared by myself and without any unauthorized help or assistance. Only the specified sources (references, tools, etc.) were used. All parts taken from other sources word by word, or after rephrasing but with identical meaning, were unambiguously identified with explicit reference to the sources utilized.

I authorize the Faculty of Electrical Engineering and Informatics of the Budapest University of Technology and Economics to publish the principal data of the thesis work (author's name, title, abstracts in English and in a second language, year of preparation, supervisor's name, etc.) in a searchable, public, electronic and online database and to publish the full text of the thesis work on the internal network of the university (this may include access by authenticated outside users). I declare that the submitted hardcopy of the thesis work and its electronic version are identical.

Full text of thesis works classified upon the decision of the Dean will be published after a period of three years.

Budapest, 16 December 2023

.....  
Mohammaderfan koupaei

# Summary

In this thesis, the relation between music composition and artificial intelligence (AI) is explored. Music has historically been challenged by societal changes, dividing into diverse genres and styles. Currently, AI's rapid advancement raises questions about its potential in music creation. This research investigates the capability of computers to capture the emotional and creative depth of human-composed music, focusing on the conversion of lyrics into melodies.

The study is driven by a deep interest in both computer science and music. With machine learning's growing influence, this thesis examines whether AI can replace or compete with human creativity in music composition. The research aims are firstly, addressing the challenges of using AI to create melodies from lyrics and suggesting potential solutions; secondly, discussing the creation of a MIDI file dataset designed for this purpose; and finally, training a model to transform lyrics into melodies. The evaluation of the model's performance and its ability to mimic human-made music is the core focus.

Commencing with a review of existing literature in the field, the thesis discusses challenges and potential improvements. It then details the data collection and processing efforts aimed at optimizing model performance. The development, training, and final evaluation of the model in both qualitative and quantitative phases are thoroughly covered, providing insights into the intersection of AI and music composition.

## نگاهی کوتاه

در این پایان‌نامه، من به طور عمیق به ترکیب جالب توجه ترکیب‌سازی موسیقی و هوش مصنوعی می‌پردازم. موسیقی، در طول تاریخ، همواره یک هنر در حال تغییر بوده است، که با جامعه تکامل یافته (AI)، ما نقش بالقوه آن در AI و به وجود آوردن ژانرها و سبک‌های بی‌شماری شده است. اکنون، با پیشرفت سریع خلق موسیقی را مد نظر قرار می‌دهیم. این پایان‌نامه بررسی می‌کند که آیا کامپیوترها می‌توانند به عمق احساسی و خلاقیتی که در موسیقی ساخته شده توسط انسان‌ها یافت می‌شود، دست یابند، به ویژه در تبدیل شعر به ملودی.

این مطالعه توسط علاقه‌مندی عمیق به هر دو رشته علوم کامپیوتر و موسیقی جمع‌آوری شده است. با توجه به اینکه یادگیری ماشینی به طور فزاینده‌ای در شکل‌گیری آینده ما نقش دارد، این پروژه به دنبال درک این می‌تواند به طور مؤثر جایگزین یا بهبود بخشیده خلاقیت انسانی در ترکیب‌سازی AI موضوع است که آیا برای ایجاد ملودی‌ها از AI موسیقی شود. اهداف این پایان‌نامه متنوع است. ابتدا، به چالش‌های استفاده از طراحی MIDI شعرها نگاه می‌کند و راه‌حل‌های احتمالی را پیشنهاد می‌دهد. سپس ایجاد مجموعه داده‌های فایل شده برای این منظور را مورد بحث قرار می‌دهد، که از آن پس با آموزش مدلی برای تبدیل شعر به ملودی ادامه می‌یابد. هدف نهایی ارزیابی کارایی مدل و بررسی این است که آیا می‌تواند سبک موسیقی ساخته شده توسط انسان‌ها را تقلید کند.

این پایان‌نامه با بررسی تحقیقات موجود در این زمینه آغاز می‌شود، که چالش‌ها و راه‌های بهبود آن‌ها را مورد بحث قرار می‌دهد. سپس به جمع‌آوری و پردازش داده‌ها می‌پردازد، با هدف دستیابی به بهترین عملکرد مدل. توسعه مدل به طور مفصل توضیح داده می‌شود، که از آن پس با آموزش و ارزیابی نهایی آن ادامه می‌یابد.



# 1 Introduction

The evolution of music throughout history is a fascinating subject for many historians. Music, as an art form, has continually evolved and reshaped itself in alignment with societal changes. Over time, countless genres and styles have generated, making music an integral part of human life across various cultures and countries. In the modern era, the rapid growing of artificial intelligence (AI) and machine learning technologies raises the question: Is it time for humans to allow computers to compose and design music for them?

A key concern in this area is whether computers can match humans in music composition, especially considering that the quality of music often lies in its emotional impact. How can these emotions be effectively transferred into a machine, and is it practical to strive for such an achievement? To date, there have been some successful models that can generate music based on specified emotions, genres, and styles. However, a model that can compose music directly from lyrics is yet to be developed to reach human level.

## 1.1 Motivation

The field of machine learning is advancing at rapidly that is out of the ability of many to reach. My interest in computer science improved curiosity and a desire to explore this rapidly changing landscape. This, combined with my passion for music and creativity, led me to enter this topic. This project aims to explore whether machines and AI can, in any sense, replace humans in the creation of music.

## 1.2 Objectives and questions

This thesis aims to address several key questions: Can AI compose music that closely resembles human creativity? How reliable is new technology in this field? What are the limitations and challenges involved? The thesis will outline a roadmap for the task of generating melodies from lyrics. The main objectives are:

- I. What are limitations for generating melody from lyrics and how to resolve this issue?
- II. Creating a dataset of MIDI files suitable for corresponding model and explaining procedure for this purpose

- III. Training a model to generate melody from lyrics.
- IV. Analyzing how effective this model is and how close to human composed style.

### **1.3 Thesis structure**

The thesis begins with a review of existing works in related fields, discussing their challenges and potential improvements. It then covers data collection, addressing availability and potential difficulties. The focus shifts to data processing, detailing optimal performance strategies and model design. The thesis then delves into the training process and concludes with an evaluation of the model's performance. The structure is as follows.

- I. Review of related literature and data gathering.
- II. Data collection and processing.
- III. Model design and development.
- IV. Testing and accuracy assessment of the model.

## **2 Related works and literature review**

This chapter intended to give a brief overview of melody generation from lyrics to readers and will cover topics 3.1-melody line identification will expand several studies on how a melody is identified in a song in both art point of view and music technician and how computation algorithms will help. Section 3.2-lyrics to melody generation dives into several projects that covered this topic.

### **2.1 Melody line extraction**

In most of music concepts melody line is primary and most important line of notes in a music however identifying this without listening can be challenging and machine learning techniques might be considered for such identifying on the other hand such understanding of melodies is important for both musical features understanding and musical based application to achieve highest performance. Nowadays musicians use new methods to produce their piece of art and majority of them are using MIDI files to illustrate their music core and concept.

Midi files (Musical Instrument Digital Interface) are described a set of musical symbols representation known as score and they are structured in multi-track level and only one is considered as the melody. There have been numerous researches on identifying melody in a multi-track MIDI file where study by David Rizo [1]2.1 have introduced method by extracting set of descriptor from each track of the target file and descriptors are input to a random forest classifier that assigns the probability of being melodic line. A novel melody line identification algorithm for polyphonic MIDI files is published [2] that A note pruning, and track / channel method is used to identify melody and has achieved 97% accuracy in identifying melody line. Using Skyline algorithm is a popular method in extracting melody line throw MIDI file by the study [9] clusters midi channel depending on the pitch histogram and a channel is selected and skyline algorithm will be applied to ensure accuracy. Another study [7] leveraging on two stage 1-main melody classification using MIDIXLNet model and 2-conditional prediction using a modified MuseBERT model.

As most of extraction algorithms more focus on skyline algorithm and probability of being melody line [13] introduce a naïve bayes classifier and probabilistic graphical

model (PGM) is designed to extract melody vectors of music features from each track of MIDI file but in a study from [3] is a dictionary based algorithm to extract melody from numerical musical scores by direction of intervals instead of notes where can lead to more flexibility for adopting such algorithm on data.

In context of music structure and understanding melody in its artistic way [4] has explained properties of music and perception and solutions for monophonic melodies extraction it is also crucial to analyse music in regards to its meaning and emotion where emotions effect on melody and temperature of music [6] has proposed that how melody can be sensed due its impact on human.

Studies [8]and [11] shows how human can distinguish different levels of sounds and voices in a music where [11] says “A single 'voice' may consist of one or more synchronous notes that are perceived as belonging to the same auditory stream” showing how identical music notes in different level can be for listeners.

In the methodological way of applying melody extraction out of a midi file knowing only probabilistic algorithms and human feelings is not sufficient and machine learning approaches must have been taken into account [10] has done a significant research from voice level separation and melody detection using CNN and modeling a musical score ,this research has broken down a music score into piano roll pieces and using CNN have trained a model to refine a score into monophonic and single track described as melody , another machine learning method has tested by [5] that can identify accompaniments and melody line in imbalanced scenarios by track feature extraction using classifiers and resampling methods .

A big challenge to address this issue and solving using neural networks is to label data , as far as there is no database that has separated melody and accompaniment[12] have annotated track into melody and accompaniment , applying multiple regression and support vector machines on features directly or on a gaussian transformation of the features resulted a model of 90% accuracy in predicting correct melody.

## **2.2 Lyrics to Melody Generation**

By exploring topic of auto melody generation from lyrics it is obvious that this field has faced rapid change in its algorithms and computation techniques [14] music

generation is being known as an art by relying on knowledge and experience artists had attempted to address this issue , since AI revolution and its impact on today life melodies were written and generated purely by artists.

As an entry point Torsten and Eduardo in their study [16] have mentioned how constraint programming is effectively used for modeling various music theories, but for complexity of this field many researchers have tried to introduce multilevel models for composing melodies such as [17] with a hierarchical model and [18] focuses on using neural networks to identify and learn the unique structural elements of a composer's style.

While these models attempt to compose a MIDI file layer by layer starting from pitch or tempo and appending other musical features above it , single error in any stage could lead a composed file that has no similarity to human version , for that many studies focus on training a model to generate a unique musical element and they construct MIDI file based on that output such as [19] trained a model generating MIDI files by integrating music theory principles and structuring compositions from RNN-predicted segments and Popsong framework [20] generating lead melodies and piano accompaniments using chord progressions, using three separate models, altering chord progressions for style, generating melodies using seasonal ARMA processes, and integrating these melodies for piano accompaniment.

Work of Watanabe [21] which is a melody to lyrics generation completely clears the complexity of this field its mentioned that lack of aligned data between lyrics and melody is main issue that several models have failed to replace human composing style , to resolve aligned lyrics and melody issue Telemelody[22] had tried to reduce this dependency by introducing a two stage model from lyrics to a template and template to melody using phonemes[23] and rhythm pattern , ROC [23] is another study pointing out that lack of aligned melody and lyrics makes it difficult to train a neural network model for melody generation , they have setup a 4 layer decoder only and using a database including verse and chords lyrics and their probable melody notes try to predict melody from lyrics from user defined chord progressions. Two other works Meloform [26] and RelyMe [25] have mentioned how lack of aligned data has effected and try to generate melodies using an expert system which is designed to generate melodies from motifs to sections, following a pre-defined musical form, and a Transformer-based model is employed to enhance musical richness without altering the form.

### **3 Background knowledge**

Traditional area of art and more specifically music is piece of attention and detailed explanation in this thesis , beside technical aspects of this task ,without sufficient knowledge in music field it's not possible to reach favorable results , no need to mention that music and melody composing is not only a technique to learn but it needs deep understanding of emotion , rhythm ,different tones employing different instruments and putting all together must result in a smooth and feelable piece of art

In this section I am going to explain knowledges gained throw this process and mention their usage in related task , this section is present in three main parts first monitoring the music theory and multiple elements of it that must be well understood to apply in final machine learning model at second part I am going to expand details of lyrics and its core relation to melody and how to achieve a smooth and human desirable melody from lyrics and finally technical aspect and knowledges used in this work is explained in details

#### **3.1 Music theory**

Music theory encompasses the study of music's practices and possibilities, as detailed by The Oxford Companion to Music[27], which identifies three aspects: understanding basic music notation (like key and time signatures), exploring historical scholarly perspectives on music, and a musicology sub-discipline focusing on defining music's general processes and principles, distinct from music analysis by its focus on music's foundational elements.

For more illustrating music notation used in this thesis, detailed explanations of them are provided and famous song “Twinkle Twinkle” is used as example for better visualization of these elements.

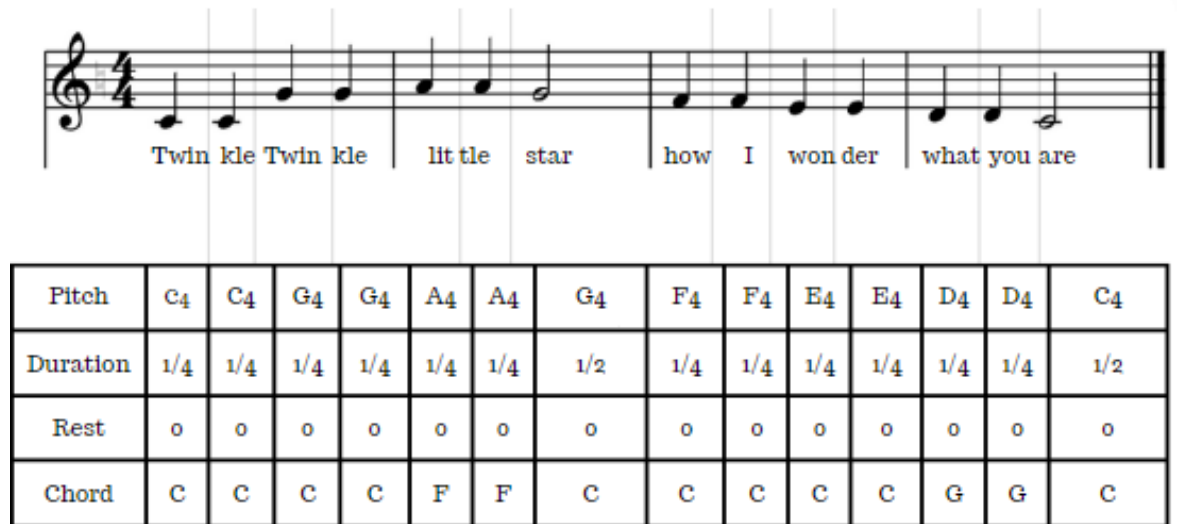


Figure 3.1 Music score and related elements

### 3.1.1 MIDI file

A MIDI (Musical Instrument Digital Interface) file is like a digital sheet of music. Unlike traditional audio files that record the exact sounds, a MIDI file saves the musical notes and instructions for playing them. It's like a recipe for music: it tells you what notes to play, how long to hold them, and how loudly to play them, but it doesn't contain any actual sound. This makes MIDI files incredibly useful in music research and digital composition. Researchers can tweak each note and see how it affects the whole piece, which is great for studying music theory or creating new compositions with computers. It's a powerful tool because it turns music into data that can be analyzed and manipulated with precision, offering a unique window into the mechanics of music.

### 3.1.2 Pitch

Pitch is a perceptual property of sounds that allows their ordering on a frequency related scale [28] . In our computational models, pitch will be represented numerically, corresponding to the specific notes in a MIDI file. In the context of "Twinkle Twinkle Little Star," in Figure 4.1 pitches are C<sub>4</sub> (middle C), G<sub>4</sub>, A<sub>4</sub>, F<sub>4</sub>, E<sub>4</sub>, and D<sub>4</sub> correspond to specific notes on the musical scale.

Figure below shows how pitches will look in a music sheet according to their high or low frequency.

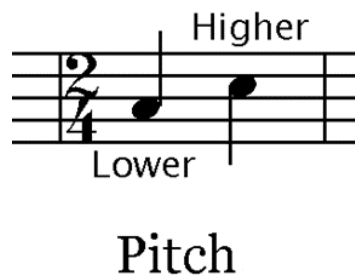


Figure 3.2 pitch frequency[30]

### 3.1.3 Duration

Duration refers to how long a note is held in a piece of music. It contributes significantly to the rhythm of the song. In computational terms, duration can be represented in various ways, such as the number of beats a note is held the duration of each pitch in the melody is measured in beats; in "Twinkle Twinkle Little Star," most notes are a quarter note in duration, which means each note is held for one beat. There are also half notes, which are held for two beats, providing a longer sound, and contributing to the song's rhythm.

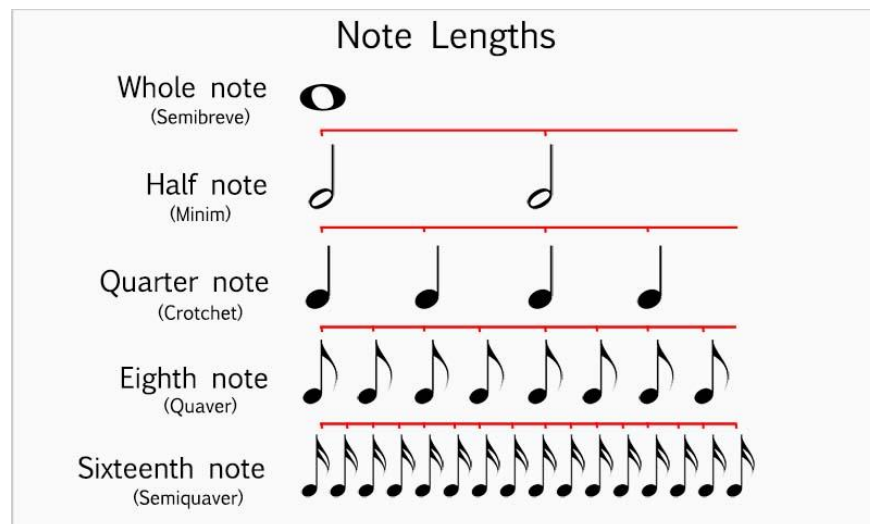


Figure 3.3 Notes duration sign [29]

### 3.1.4 Rest

Rests are symbols in written music that indicate silence or the absence of sound for a specific duration. They are as essential to the rhythm and feel of a piece as the notes themselves. There are several types of rests, each corresponding to a note value. Here is a brief overview of common rest durations:



- I. **Whole Rest:** A whole rest lasts for four beats in common time (4/4-time signature). It is denoted by a filled-in rectangle hanging under the second line from the top of the staff.
- II. **Half Rest:** A half rest is worth two beats in common time. It looks like a filled-in rectangle sitting on top of the third line of the staff.
- III. **Quarter Rest:** This rest gets one beat in common time and is represented by a squiggly vertical line.







0	1	2	4	8	16	32
No rest		 (half rest)	 (whole rest)	2 × 	4 × 	8 × 

Figure 3.4 Relationship between rest values and corresponding symbols[29]

### 3.1.5 Tonality

refers to the system in which music is composed around a central note, scale, or key, giving a sense of hierarchy and order to the notes in a piece. The circle of fifths shown below is a visual representation that helps musicians understand how different keys are related within this system of tonality. It shows the relationship between major and minor keys and how they are interconnected through their scales. The concept of a "related key" within this system highlights keys that share many common notes, making them tonally closer.

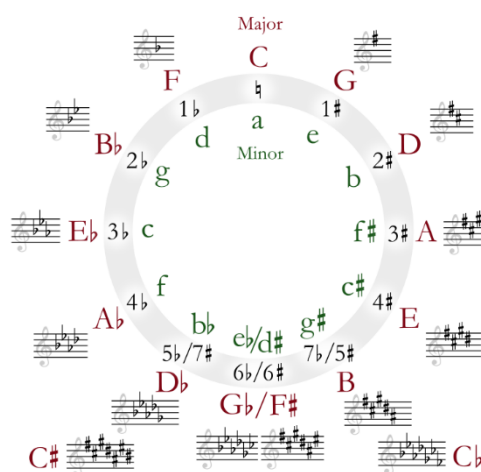


Figure 3.5 Circle of fifth

### 3.1.6 Chord progression

Is defined as a sequence of chords played in a piece of music, where the succession of these chords gives the piece its harmonic structure and character in figure below, he chords progression is "I-IV-V-IV-V-I" ("C-F-G-F-G-C" in C major scale)



Figure 3.6 chord progression

## 3.2 Lyrics and melody

In popular music, lyrics and melody are blended by composers to create an emotionally engaging piece. Understanding how these elements interact is important for developing the song's message and can influence various applications in music technology and creative assistance.[32]

In this study, lyrics are the primary input from which the melody is generated. For better understanding lyrics I must break down lyrics into word and syllable and phoneme level and where each of them plays a crucial role for understanding music structure and rhythm for creating a coherent musical piece and then explaining how machines can understand human words in context of natural language processing.

### 3.2.1 Syllable and phoneme

In language, a syllable is like a beat in a word made up of vowel and consonant sounds; it's one piece of the word that you can hear. A phoneme, on the other hand, is the tiniest sound in a language that can change the meaning of a word[34].

For example, in the English word "twinkle," the two syllables [twɪn] and [kəl] are composed of five phonemes (i.e., /t/, /w/, /ɪ/, /n/, and /kəl/) for breaking down words into syllable level following steps must be taken:

**-Identify Vowel Sounds:** Every syllable must have at least one vowel sound (like /a/, /e/, /i/, /o/, /u/).

**-Find Consonants:** Look for consonants that come before or after the vowel sounds. These often help define the syllable boundaries.

**-Divide the Word:** Split the word at the point where it naturally breaks according to the vowel and consonant sounds, ensuring each syllable has at least one vowel sound

"twinkle" is divided as /'twɪŋ.kəl/ or [twɪn][kəl]. The first syllable /'twɪŋ/ (twin) includes the consonant sounds /t/ and /w/ followed by the vowel sound /ɪŋ/ (short 'i' as in "win"). The second syllable /kəl/ (kəl) begins with the consonant sound /k/ followed by the vowel sound /əl/.

### 3.2.2 Natural Language Processing

**NLP[35]** plays an important role in processing lyrics within the context of music generation and analysis. NLP technologies enable machines to understand, engage, and generate human language in a way that is meaningful for musical applications. This is significant in the field of automated music composition and lyric analysis.

As mentioned in 2.2 lack of aligned melody and lyrics always pushed researchers to find alternative solutions , but for reaching level of human style composing melodies can't deny role of lyrics and its relationship to melody , in favor of NLP technology its more and more possible to align melodies and lyrics and in this thesis I have tried to employ different techniques for best performance and result such as NLTK and Tokenizers and Word2Vec model

#### 3.2.2.1 NLTK

The Natural Language Toolkit (NLTK) is a popular Python library that provides tools for working with human language data (computational linguistics and natural language processing). It includes support for tasks such as text processing, linguistic analysis, and language modeling.

#### 3.2.2.2 Tokenization[37]

Procedure that breaks down text into smaller units, such as words, phrases, or sentences. These smaller units, often called tokens, are necessary for deeper language analysis or for feeding into machine learning models for tasks like language translation, or melody generation from lyrics. By splitting text into tokens, a tokenizer helps the algorithm to process language in a structured and quantifiable manner.

Total Tokens	967154
Unique Words	15173

Total Lyric Entries	216809
Average Tokens per Entry	4.460
Total Melodic Entries	216809
Average Pitch Variations	9.038

Table 3.2.2.2 distribution of tokens in dataset

### 3.2.2.3 Word2Vec

Word2Vec[38] is a machine learning model that transforms words into numerical form, specifically into vectors of real numbers, so that the computer can understand and process them. In the context of this thesis, Word2Vec can be used to convert lyrics into a format that captures not just the words themselves, but also their meanings and the relationships between them. In this thesis I have used Skip-Gram word2vec model

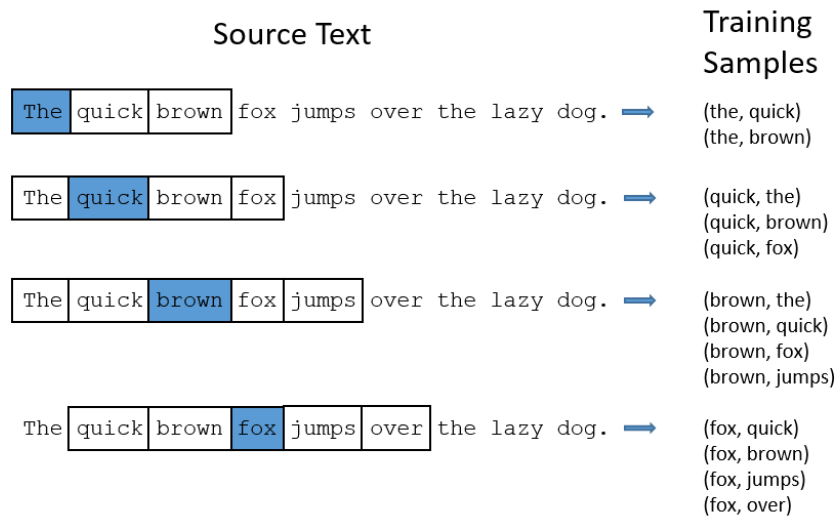


Figure 3.8 skip-gram model visualization

The Figure 3.1 explains the skip-gram model in Word2Vec, where for a given target word, the model predicts surrounding context words. For instance, with "quick" as the target, the model aims to predict "The," "brown," "fox," and possibly "jumps" as its context, demonstrating how words are expected to co-occur. In the context of this thesis, this model can be applied to understand and capture the contextual relationships between

words in lyrics. By doing so, can inform the system about which notes and melodies best align with certain words or phrases.

### 3.3 Technical overview

In this section, I will explain the application of machine learning models, specifically sequence-to-sequence and transformer models, in the context of music generation from lyrics. These models are designed to receive sequences of input and outputs and learn how they are related to , in scope of lyrics to melody , lyrics can be served as input and melodies as output in sequences.

#### 3.3.1 Transformers

Transformer[39], introduced in 2017, are a breakthrough in deep learning for efficiently processing sequential data. Unlike earlier recurrent neural networks like LSTMs, transformers work in parallel processing, significantly speeding up training times. They work by first breaking down input text into n-grams, converting these into vectors using a word embedding table. The major benefit is in their multi-head attention mechanism[39], which processes all tokens simultaneously.

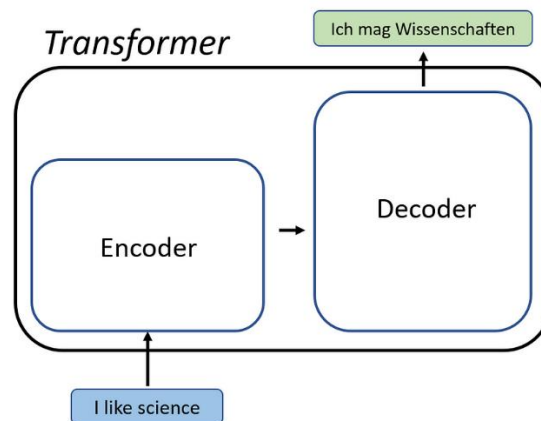


Figure 3.2 Simple transformer architecture

#### 3.3.2 Sequence-to-sequence Transformer model

sequence-to-sequence (seq2seq) models, essential in machine learning for converting one sequence into another, like turning text into a corresponding translation. These models have two main parts: an encoder, which transform the input into a compact form, and a decoder, which then recreates a new sequence from this transformed form. In

music generation, seq2seq can transform lyrics into melodies, making it a powerful tool for linking words with music.

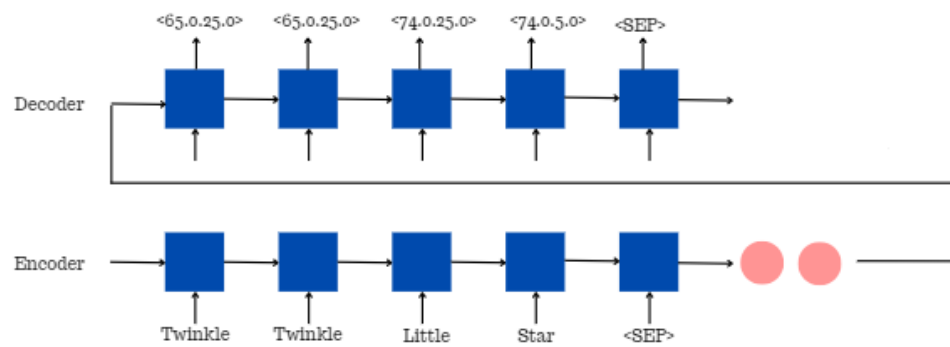


Figure 3.3 seq2seq transformer architecture

## 4 Methodology and model design

Process for designing majority of transformer models is divided into three main parts.

- I. data collection and preprocessing
- II. training
- III. evaluation and result.

This chapter is focused on data collection and preprocessing and training phase of the thesis.

### 4.1 Data collection and preprocessing

First step to train a lyric to melody model is gathering a MIDI files dataset that supports vast number of lyrics and musical genres, then collected data must preprocess according to goal of task and feed into model. Here I will first explain dataset used and procedure to apply refinements and filtering to access a suitable format of data for my desired model.

#### 4.1.1 Dataset

There are numerous MIDI files dataset published , but for aim of this thesis I have choose LAKH[42][41] midi dataset with LAKH MIDI piano roll[45] dataset

LAKH MIDI dataset is the full collection of 176,581 deduped MIDI files and has multiple subversions which I used LMD-matched - A subset of 45,129 files from LMD-full which have been matched to entries in the Million Song Dataset[43] using checksum file that metadata of each track can be extracted.

#### 4.1.2 Preprocessing

After collecting data from LMD matched dataset, in preparation phase of data I will introduce a pipeline of three main stage.

- I. filtering data
- II. melody line extraction and processing lyrics and melody
- III. lyrics and melody alignment

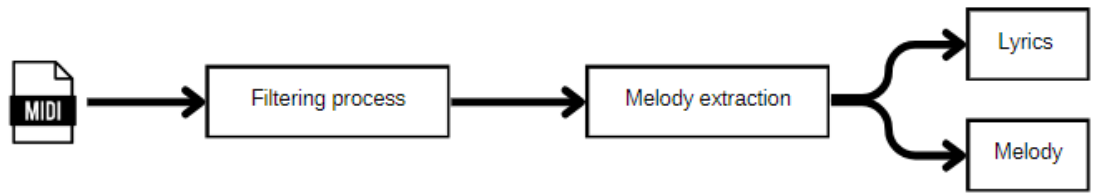


Figure 4.1preprocessing pipeline

#### 4.1.2.1 Filtering data

LMD matched dataset consist of more than 45000 files and I needed to pick most suitable files for my model , in according to achieve this first stage was removing songs that are less than one minute length because they don't contain diverse melody and lyrics and might end up failure for model parallel to this level by using Music21 library and time signature detection feature only 4/4 time signature songs were taken into account for rest of filtering phase as this time signature is most popular in western melody composing and also makes lyrics melody alignment step more manageable, second filter applied was checking songs whose lyrics are attached to file , this is necessary due to multiple versions of song are available and determination of the matched one with MIDI file is not possible for checking availability of lyrics I used pretty-Midi library to detect presence of lyrics in files , this stage was parallel done with checking whether the detected lyrics are English , with employing NLTK library and BART model for separating English and non-English songs , this method is not completely accurate due to some songs may contain some words that are similar to English but they consider non – English words or some Midi files contained some metadata that were in English but context of song was another language to resolve this issue I set constraint of 60% for  $N(\text{English words}) / N(\text{total})$  to ensure all midi files are in English.

#### 4.1.2.2 Melody line extraction

As totally explained on 2.1 Standard MIDI files encapsulate various musical data elements, such as note onsets and offsets, pitch and rest, and durations, collectively forming a music score. These scores are typically organized into multiple tracks, each representing a different instrument, with one distinct track dedicated to the melodic line, while the others serve as accompaniments. In this phase of the thesis, pattern recognition techniques were employed to identify the most probable melodic track among the others. It's important to mention the importance of data normalization during model training, especially in the context of aligning lyric timestamps with musical features. The presence



of multiple tracks and instruments in a MIDI file can lead to a huge data load that may not be essential for the model, potentially causing information loss. Moreover, my primary objective is to generate a melody based on lyrics, makes me focus on the melodic. to explain how melody line was extracted first I must explain a MIDI score as a piano roll where notes are falling for singer to play.

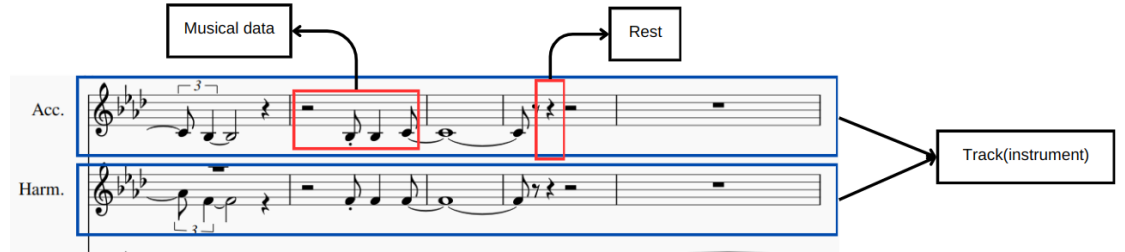


Figure 4.2 Midi score with two track

In figure above two different tracks(instruments) of a MIDI score are represented, There have been multiple researches on how to extract melody from MIDI file that were already explained in detail , my solution to this goal was inspiring from MusGan [44] which is a Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment and they have used lakh midi piano roll dataset were each track(instrument) in MIDI files is transformed into a single file, and by using MSD file picking up songs that have higher confidence score (assigned score in MSD checksum file to each track file in Lakh midi piano roll dataset representing accuracy of transformed notes into piano roll and matching with original model) and finally they formed a dataset of piano rolls from LMD matched dataset with highest confidence score each containing only five track 1-bass 2-drum 3-guitar 4string 5-piano.

With using piano roll dataset, I was able to pick piano roll files common with my filtered dataset and as its mentioned MuseGan in 96% of occurrences melody line is piano track in their dataset.

At end of this phase I had a dataset of MIDI files with lyrics and corresponding melody line

#### 4.1.2.3 Lyrics preprocessing

The initial step in lyric processing involves tokenization, which is critical for reaching a refined set of words. This process is necessary for producing a diverse

alignment between lyrics and melody, resulting for effective model training. Parallel to word-level tokenization, the lyrics are further broken down into syllables. This approach is used to keep highest level of consistency between aligned melody and lyrics where each stress in syllable will form melody body structure.

To achieve a deeper understanding of the stress patterns in language, the concept of phonemes is introduced. This phoneme analysis aids in identifying the end of sentences or sequences within the lyrics. Given that the focus of the research is on training a sequence-to-sequence model, it becomes more important to prepare datasets that meaningfully correlate sets of words with their corresponding melodic sequences.

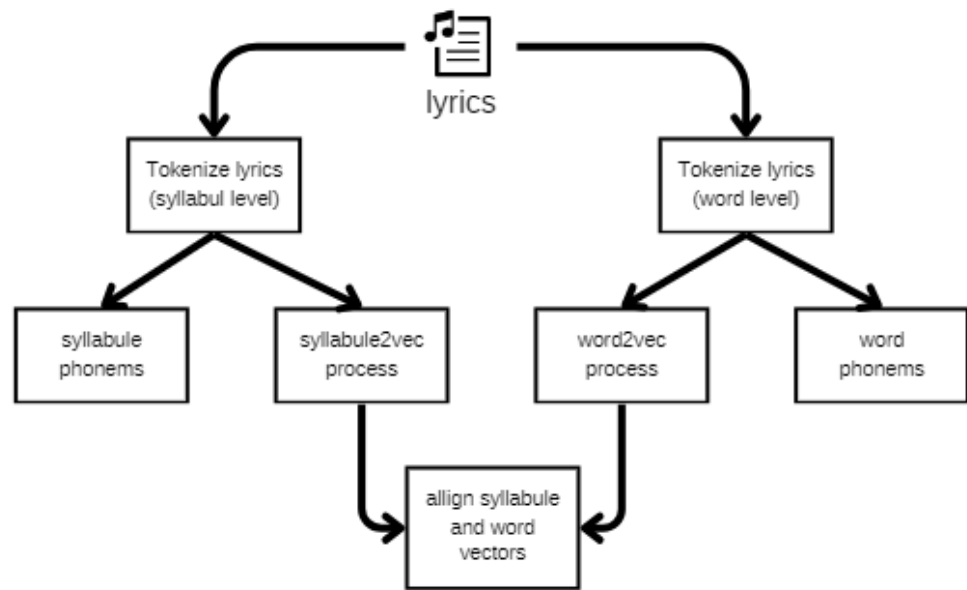


Figure 4.3 lyrics processing pipe line

In the process of phoneme generation from words and syllables, the CMU Pronouncing Dictionary[49], a component of the NLTK library, is used. This library categorizes vowels with an emphasis on stress. Additionally, the occurrence of rests within a sequence is a sign of end of sentence or sequence most of times.

The study of phonemes helped me get a better understanding of rhythm patterns in music. By figuring out these patterns, it was possible to guess where sequences in the music might end. Knowing how rhythms work is key to matching lyrics with the right note, which helps train my music-generating model effectively.



$t = Noteoff - Noteon$  , where  $t$  is in seconds .and for calculating duration of each note  $d = rand \left( t \times \frac{BPM}{60} \right)$  , where BPM is a unique number in each midi file ( beats per minute) and by having a rounding constraints we can have a normal duration for each note for instance if d is 0.33 it will be considered as a quarter note as its closer to 0.25.

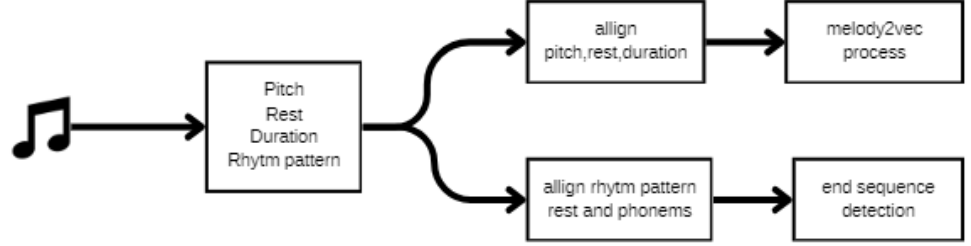


Figure 4.5 pipe line of melody processing

By extracting Pitch (midi number ) and its duration and possible Rest ( and its duration) for each note I ended up a triplet <MIDI number , duration , rest> and rhythm pattern could be conducted using BPM and midi tick number using equations bellow:

$$Beat \text{ Duration (seconds)} = \frac{BPM}{60}$$

$$Tick \text{ Duration (seconds)} = \frac{TPB}{Beat \text{ Duration}}$$

$$Note \text{ position in measure} = \frac{(Note \text{ onset Time(in ticks)} \times mod (ticks \text{ per Measure}))}{Ticks \text{ per Bear}}$$

Where note position in 4/4-time signature is a number between 1-4 where we can determine rhythm pattern.

By employing phonemes to detect syllables and Rests in midi file and rhythm pattern I could detect a phrase end in my dataset. Let  $S$  be the number of syllables,  $R$  be the occurrence of a rest, and  $P$  be the number of rhythm patterns. Then, the condition for the end of a phrase can be expressed as:

$$End \text{ of Phrase} = \begin{cases} 1 & S \geq 8 \text{ and } R \text{ occurs after the 8th syllable} \\ 0 & P \geq 2 \text{ and no } R \text{ occurs between the patterns} \\ 0 & \text{other wise} \end{cases}$$

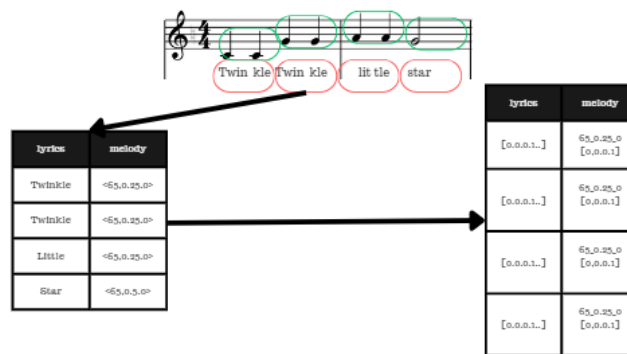


Figure 4.6 process of melody and lyrics extraction

#### 4.1.2.5 Applying word2vec

In this phase a skip-gram word2vec model was employed to transform both melodies and lyrics in vectors that more closely words assign closer vector and same for melodies it is beneficial for constructing as much as possible meaningful data passed through the model

In my word2vec dataset for instance for word “love”

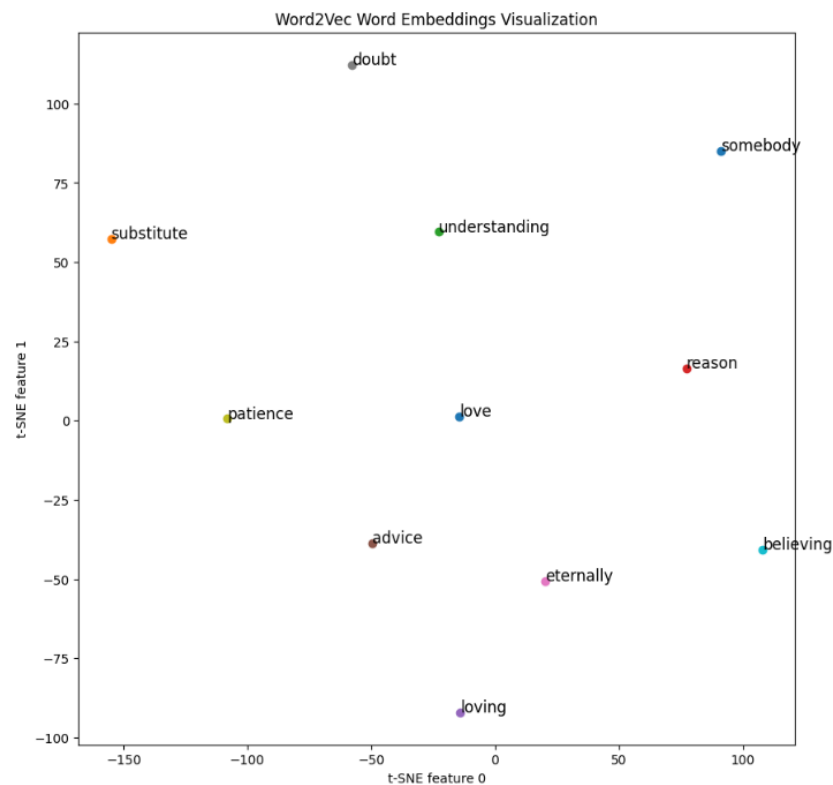
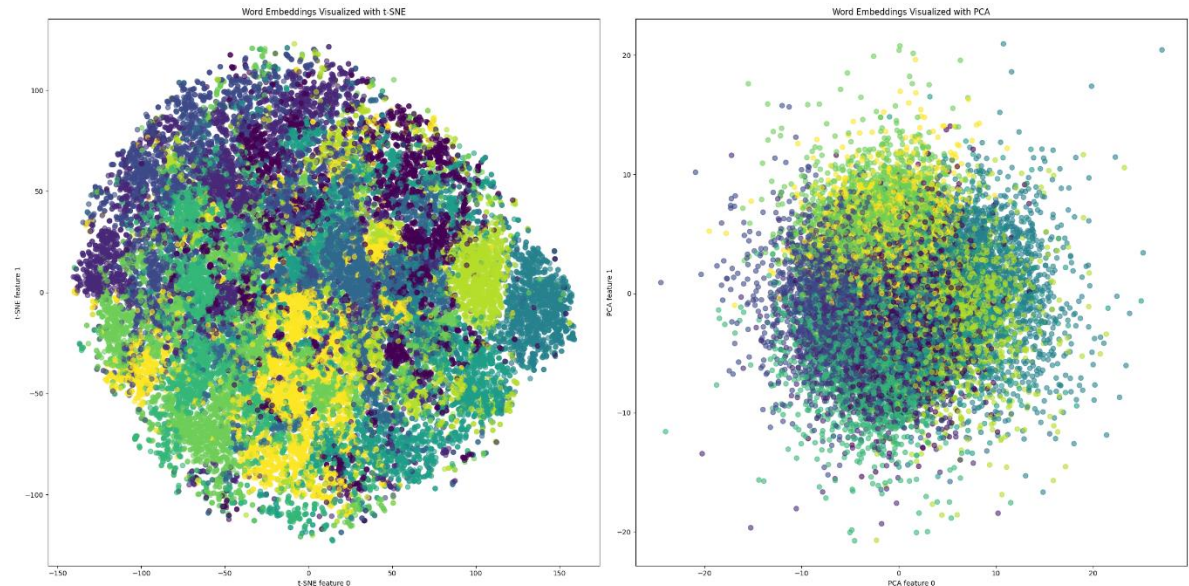


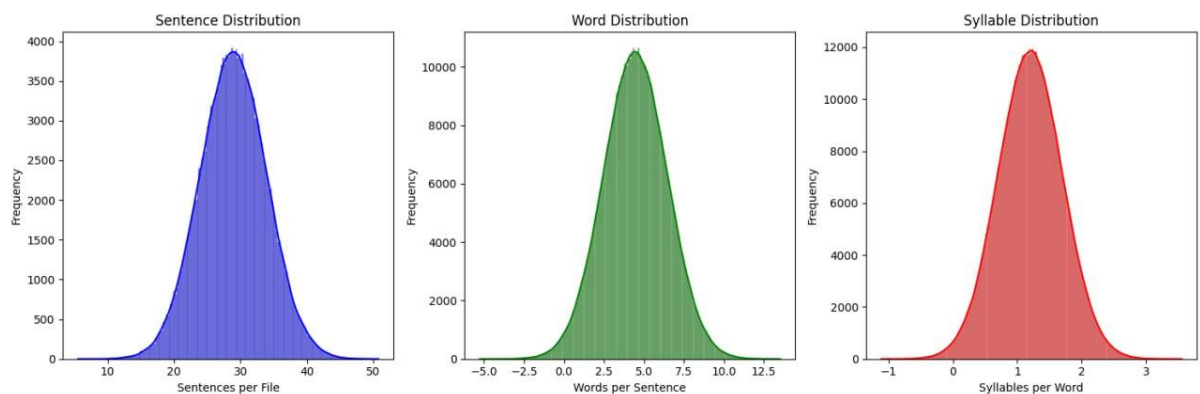
Figure 4.7 word2vec embedding for love

Words that were assigned closer meaning to it are shown, these words were structured into vectors of size 100 and as previously explained a skip-gram model applied to get more closely meaning words having closer vectors , I have chosen to use skip-gram model of word2vec as it may result in less efficient compared to CBOW, but it plays better in representing infrequent words.



**Figure 4.8 word embedding visualized with t-Sne and PCA**

The t-SNE graph shows groups of words that are like "love" in meaning, with each color representing a cluster of words that are used in similar ways. The PCA graph spreads out all the words in our dataset, including "love," along two main directions, showing us the most common patterns in how words are used. Figures below illustrate sentences in file and words in sentences and syllable in word.



**Figure 4.9 sentence , word , syllable distribution**



Figure 4.10 sentence , word, syllable boxplot distribution

## 4.2 Model design

This stage details the technical aspects of implementing and training a Transformer model specifically for the task of music generation from lyrics. I will expand the model's architecture, including its input and output structures, hidden dimensions, and attention layers, along with the various hyperparameters.

### 4.2.1 Model architecture

In this stage, I used a seq2seq[46] Transformer model, which is a modern type of neural network good at working with sequences of data. Its main strength is in understanding how different parts of a sequence are connected over long stretches, like linking lyrics to the music notes they turn into. Next, we'll give a more detailed look at how this model is put together. Figure below is an overview of seq2seq transformer model using attention mechanism.

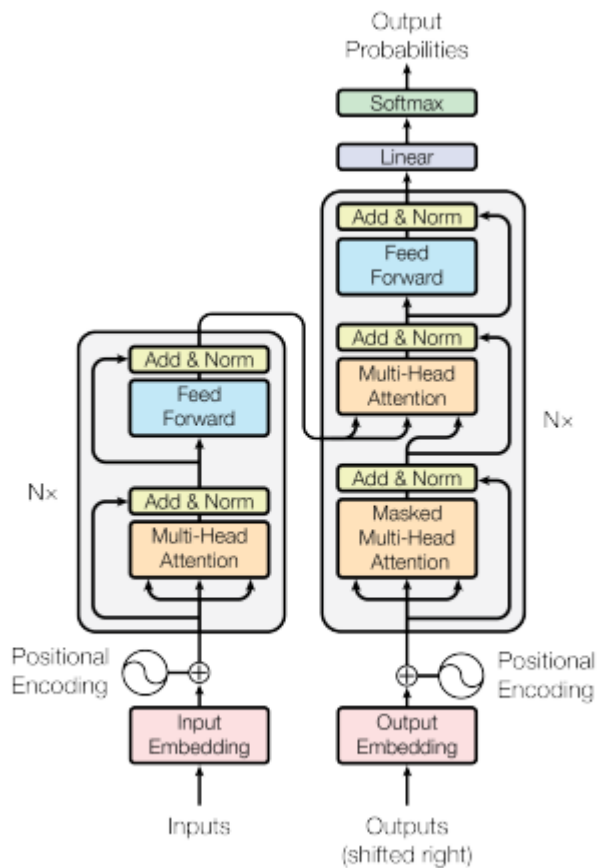


Figure 4.11 seq2seq transformer model with attention mechanism[47]

**Encoder** The encoder processes the input sequence (lyrics) and compresses the information into a context-rich representation. Each encoder layer consists of two sub-layers.

**-Self-Attention Mechanism** Allows the model to weigh the importance of different words in the lyrics relative to each other.

**-Feed-Forward Neural Network[48]:** Transforms the attention-weighted representation to a higher-level abstraction.

**Decoder:** The decoder takes the encoder output and generates the output sequence (musical notes) step by step. Each decoder layer has three sub-layers.

**-Masked Self-Attention Mechanism:** Like the encoder's self-attention but ensures that the prediction for a certain position doesn't depend on future positions.

**-Encoder-Decoder Attention:** Helps the decoder focus on relevant parts of the input sequence.

**-Feed-Forward Neural Network:** Further abstracts and transforms the data.



### Hyperparameters:

**-Input Size:** Determined by the dimensionality of the Word2Vec embeddings used for the lyrics, set at 100. This size empowers the embedded vector to capture the meaning of the lyrics.

**-Output Size:** Set to 100, matching the dimensionality of the Word2Vec embeddings for the melodies. This dimension is important for defining the complexity of the output space in terms of musical representation.

**-Hidden Size:** I choose a hidden size of 512, a balance between model complexity and computational efficiency. This size is enough to capture complex patterns without extra computational resources.

**-Number of Layers:** My model structure 4 layers in both the encoder and decoder. This number of layers allows the model to learn a more from the data.

**-Number of Heads in Multi-Head Attention:** Set to 8, this parameter allows the model to continuously focus on different positions of the input sequence, important for understanding various musical themes.

**-Dropout Rate:** A dropout rate of 0.1 is used to prevent overfitting, ensuring the model generalizes well to new, unseen data.

**-Output Linear Layer:** After processing through the encoder and decoder, the final output of the Transformer is passed through a linear layer, which projects the high-dimensional representations back to the output space size. This layer is important for generating the final sequence of music notes that correspond to the input lyrics.

**-Loss function[49]:** The Mean Squared Error (MSE) loss function was selected for this task. MSE is effective for this application as it emphasizes precise output values, making it suitable for a task where accuracy in the generated sequence is critical.

## 4.2.2 Model code

### 4.2.2.1 Utility Functions

```
def get_embedding(token, model):  
    return model.wv[token] if token in model.wv else  
    np.zeros(model.vector_size)  
  
def read_data(file_path):  
    with open(file_path, 'r') as file:
```

```

        return [line.strip().split() for line in file.readlines()]

def preprocess_melodies(file_path):
    with open(file_path, 'r') as file:
        melodies = file.readlines()
    preprocessed_melodies = []
    for line in melodies:
        notes = line.strip().split(' ')
        melody = ['_'.join(notes[i:i+3]) for i in range(0, len(notes), 3)]
        preprocessed_melodies.append(melody)
    return preprocessed_melodies

```

#### 4.2.2.2 Dataset Loader

```

class LyricsMelodyDataset(Dataset):
    def __init__(self, lyrics_data, melodies_data, lyric_model,
melody_model, max_length=300):
        assert len(lyrics_data) == len(melodies_data), "Lyrics and melodies
data must be the same length"
        self.lyrics_data = lyrics_data
        self.melodies_data = melodies_data
        self.lyric_model = lyric_model
        self.melody_model = melody_model
        self.max_length = max_length

    def pad_sequence(self, sequence, max_length):
        sequence = np.array(sequence, dtype=np.float32)
        num_padding = max_length - len(sequence)
        padding = np.zeros((num_padding, sequence.shape[1]),
dtype=np.float32)
        padded_sequence = np.concatenate((sequence, padding), axis=0)
        return padded_sequence

    def __len__(self):
        return len(self.lyrics_data)

    def __getitem__(self, idx):
        lyric_line = self.lyrics_data[idx]
        melody_line = self.melodies_data[idx]
        lyric_embeddings = np.array([get_embedding(word, self.lyric_model)
for word in lyric_line], dtype=np.float32)
        melody_embeddings = np.array([get_embedding(note,
self.melody_model) for note in melody_line], dtype=np.float32)
        lyric_embeddings = self.pad_sequence(lyric_embeddings,
self.max_length)
        melody_embeddings = self.pad_sequence(melody_embeddings,
self.max_length)
        return torch.tensor(lyric_embeddings),
torch.tensor(melody_embeddings)

```

#### 4.2.2.3 Transformer Model Class

```

class TransformerModel(nn.Module):
    def __init__(self, input_size, output_size, hidden_size, num_layers,
num_heads, dropout):
        super(TransformerModel, self).__init__()

```

```

        self.transformer = nn.Transformer(d_model=hidden_size,
                                         nhead=num_heads,
                                         num_encoder_layers=num_layers,
                                         num_decoder_layers=num_layers,
                                         dim_feedforward=hidden_size,
                                         dropout=dropout)
        self.encoder_embedding = nn.Linear(input_size, hidden_size)
        self.decoder_embedding = nn.Linear(output_size, hidden_size)
        self.output_linear = nn.Linear(hidden_size, output_size)

    def forward(self, src, tgt):
        src = self.encoder_embedding(src)
        tgt = self.decoder_embedding(tgt)
        output = self.transformer(src, tgt)
        return self.output_linear(output)

```

#### 4.2.2.4 Dataset preparation and model initialization

```

lyrics_data = read_data('processed_lyrics.txt')
melodies_data = preprocess_melodies('cleaned_melodies.txt')

# Create Dataset and DataLoader
dataset = LyricsMelodyDataset(lyrics_data, melodies_data, model_lyrics,
                              model_melodies, max_length=300)
dataloader = DataLoader(dataset, batch_size=32, shuffle=True)

# Model Hyperparameters
input_size = model_lyrics.vector_size
output_size = model_melodies.vector_size
hidden_size = 512
num_layers = 4
num_heads = 8
dropout = 0.1

# Initialize model
model = TransformerModel(input_size, output_size, hidden_size, num_layers,
                          num_heads, dropout)
criterion = nn.MSELoss()
optimizer = optim.Adam(model.parameters(), lr=0.001)

```

LAYER	INPUT SIZE	OUTPUT SIZE	NUM HEADS	HIDDEN SIZE	DROPOUT RATE
INPUT EMBED	100	512	N/A	N/A	N/A
POSITIONAL ENC	512	512	N/A	N/A	N/A
ENCODER LAYER 1	512	512	8	2048	0.1
ENCODER LAYER 2	512	512	8	2048	0.1

...	...	...	...	...	...
<b>DECODER LAYER N</b>	512	512	8	2048	0.1
<b>LINEAR</b>	512	100	N/A	N/A	N/A
<b>OUTPUT SOFTMAX</b>	100	100	N/A	N/A	N/A

**Table 4.2.2.4 Architecture of designed model**

## 5 Training

The training of my Transformer model is designed over 15 several epochs, each consisting of 6,517 steps, using the NVIDIA A100 GPU to process the data efficiently.

### 5.1 Setup

I started the training by setting up pre-trained Word2Vec embeddings and initializing the Adam optimizer, for its effectiveness in handling the learning process.

**Batch Processing:** dataset is segmented into batches, allowing the model to update its parameters incrementally and improve its accuracy in translating lyrics to melodies with each step.

### 5.2 Training Steps

**Gradient Resetting:** At the beginning of each training step, I reset the gradients to prevent interference from previous data.

**Forward Pass and Prediction:** The model takes in the lyrical embeddings and attempts to predict the melody embeddings, creating a musical output one step at a time.

**Loss Measurement:** I employ the Mean Squared Error Loss to measure the model's prediction accuracy and provide a basis for improvement.

**Backpropagation and Optimization:** The loss informs how we adjust the model's parameters through backpropagation, and the optimizer refines these parameters to decrease the prediction error.

Figures below is a simple overview of attention mechanisms.

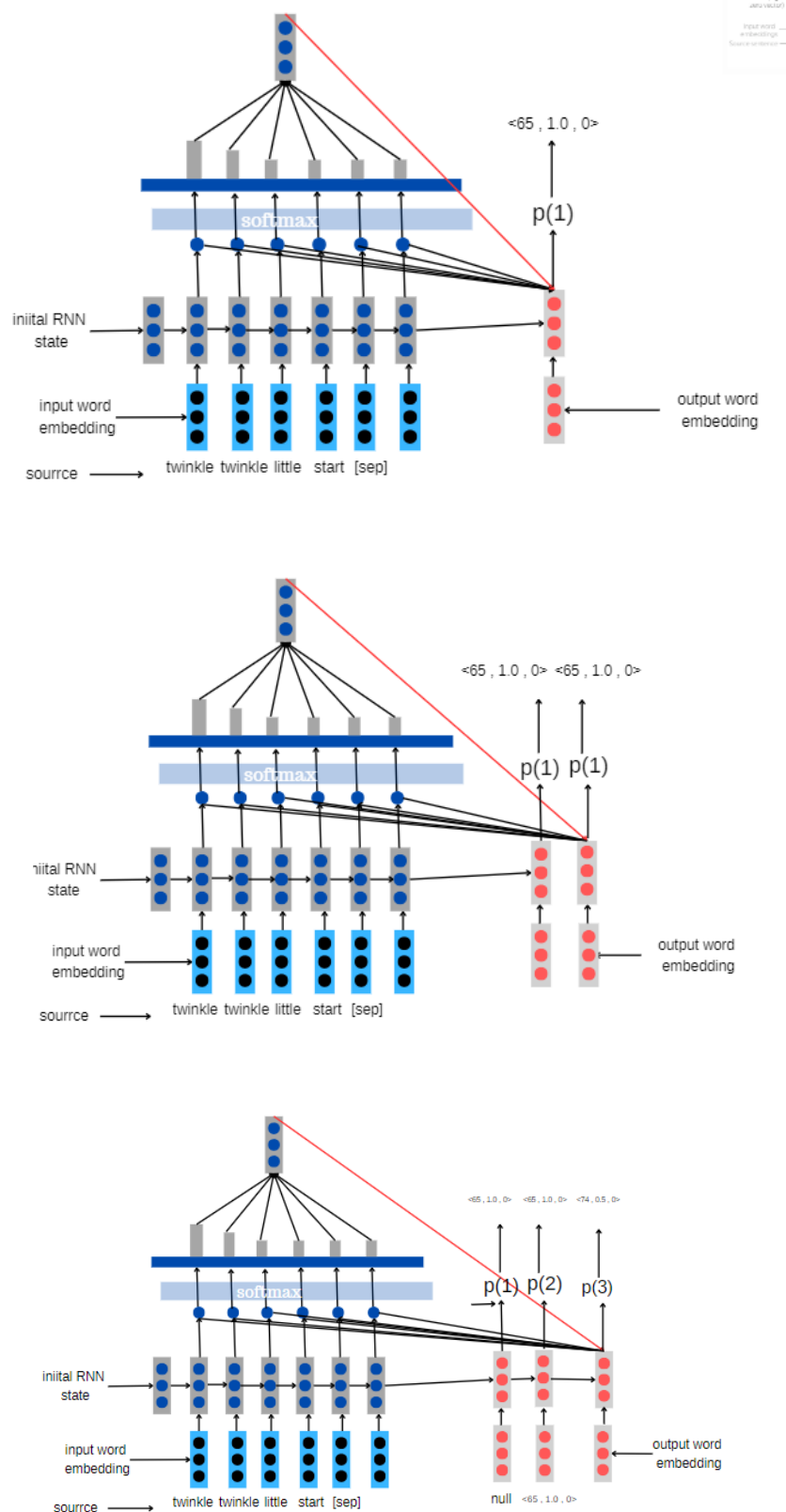


Figure 5.1 process of attention mechanism

## **6 Evaluation and results**

In this section I will examine the trained model by quantitative and qualitative means, evaluating model's performance on melodies that how well musically composed and how close they are to previously generated models and ground truth base. Quantitative analysis will capture model accuracy and precision in terms of three most important musical elements Pitch number, Duration, Rest, also will examine harmony and structure using repeated notes diversity between test data and generated melodies.

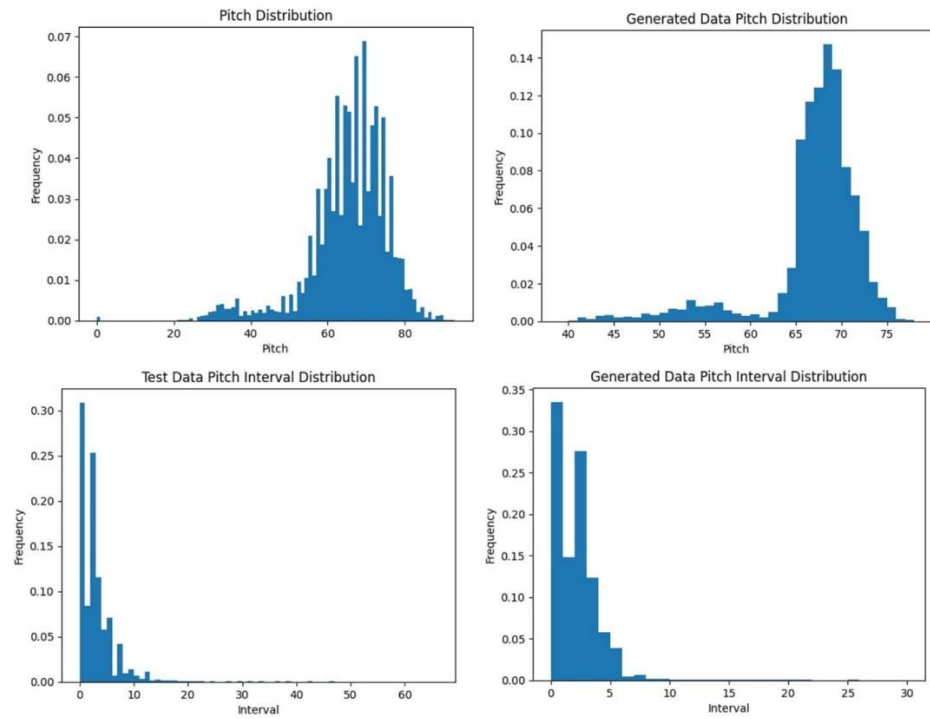
Qualitative reviews will capture human feelings related to generated melodies and using scoring system will compare.

### **6.1 Quantitative analysis**

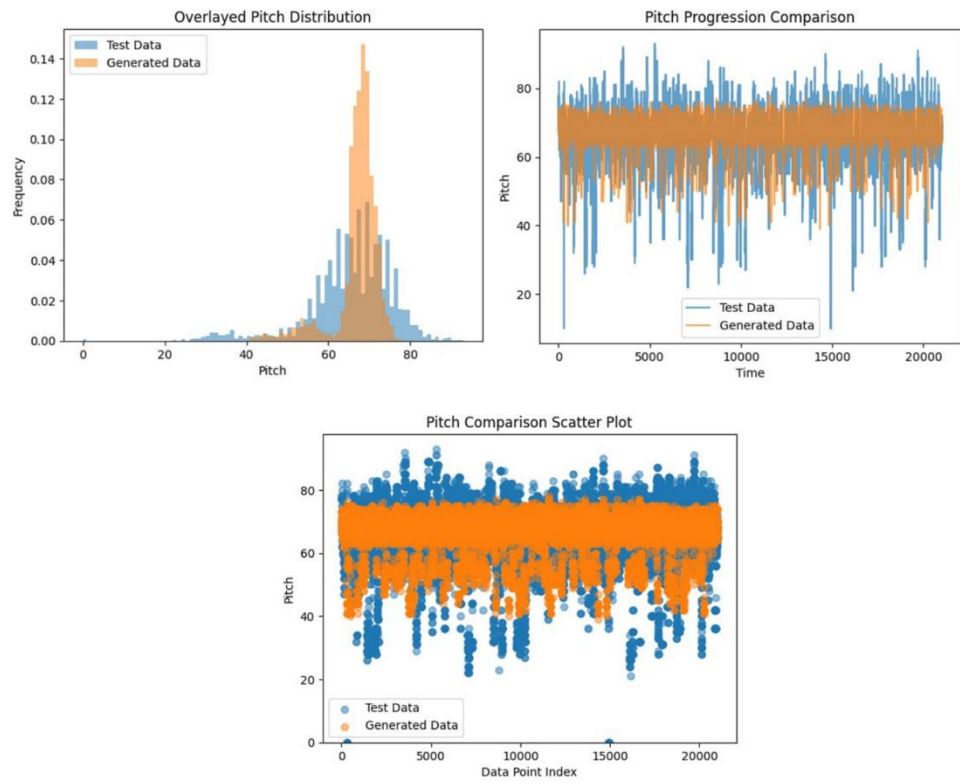
After training, the model was tested with test data to evaluate its performance in four key areas: pitch, duration, rest, and melody consistency. The results were illustrated through a series of graphs, providing a visual representation of the model's capabilities and areas for improvement.

#### **6.1.1 Pitch Analysis**

The pitch analysis, as shown in Figures 6.1 to 6.2, highlighted the model's proficiency in predicting notes within the 64 to 75 range, which are the most used in the dataset. However, the accuracy reduced for lower and higher notes. This limitation might be attributed to the number of training epochs, suggesting further training could enhance the model's pitch range accuracy.



**Figure 6.1 Distributions of rest durations and intervals**



**Figure 6.2 Various comparisons of pitch distributions and progressions**



### 6.1.2 Rest Analysis

For rest prediction, the model predominantly predicted rests as zero duration, aligning with most of the dataset. It effectively covered rest durations in the range of 0 to 5, with longer rests like 32 (eight full rests) being less accurately predicted. This could indicate a leakage in melody line extraction. The model also struggled with predicting half and quarter rests, despite their prevalence in the test dataset.

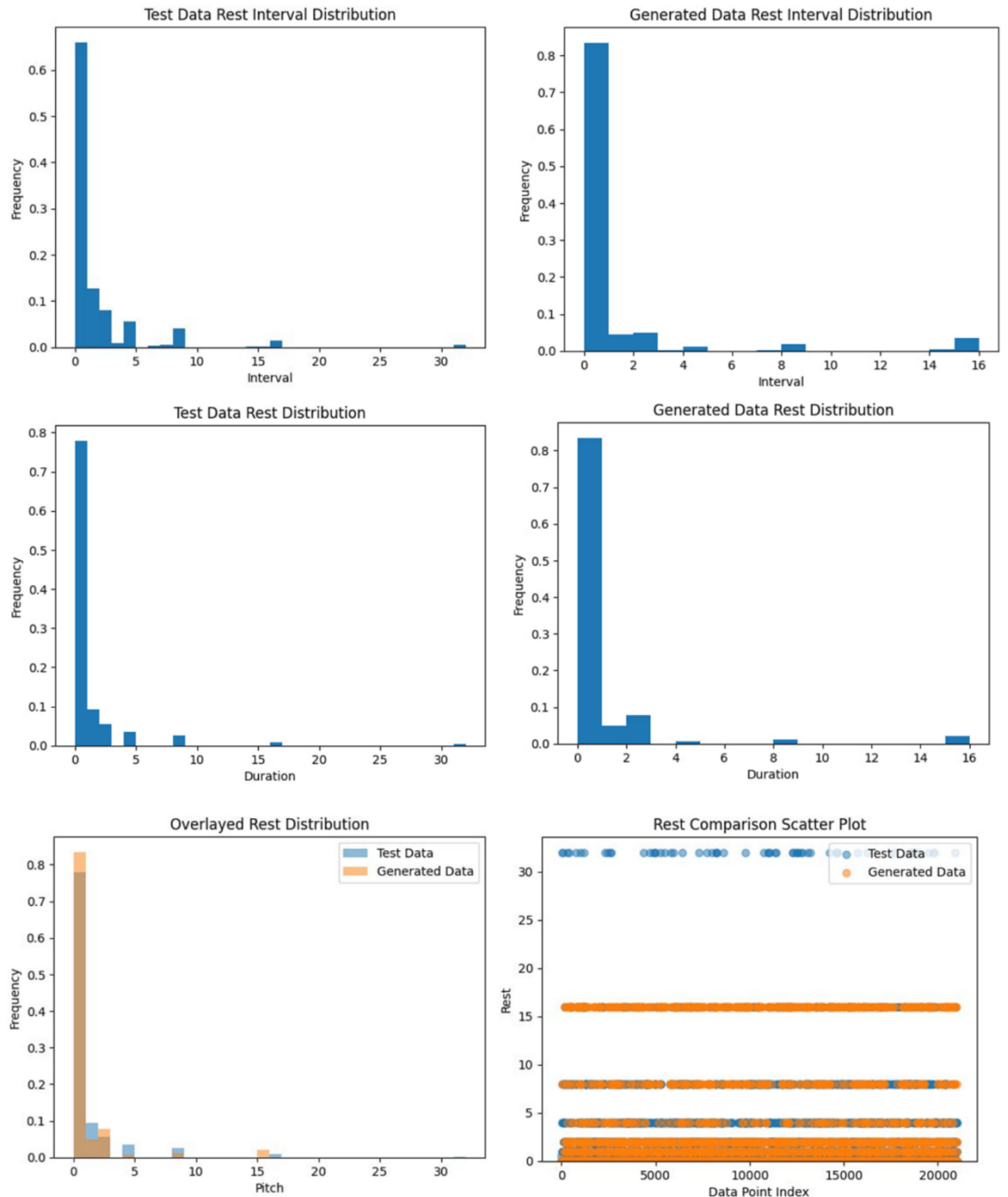


Figure 6.3 Various comparisons of Rest distributions and progressions

### 6.1.3 Duration Analysis

By analyzing pitch duration we can see that model predicted most of durations as quarter note, while dataset notes distribution is mixture of half and quarter and model couldn't well predict half notes, due to consistency model had in pitch and duration this weakness can be related to several factors most important number of trained epochs and then mistake in melody detection, results in durations is not well as Rest and Pitch when 80% of predicted durations are quarter note.

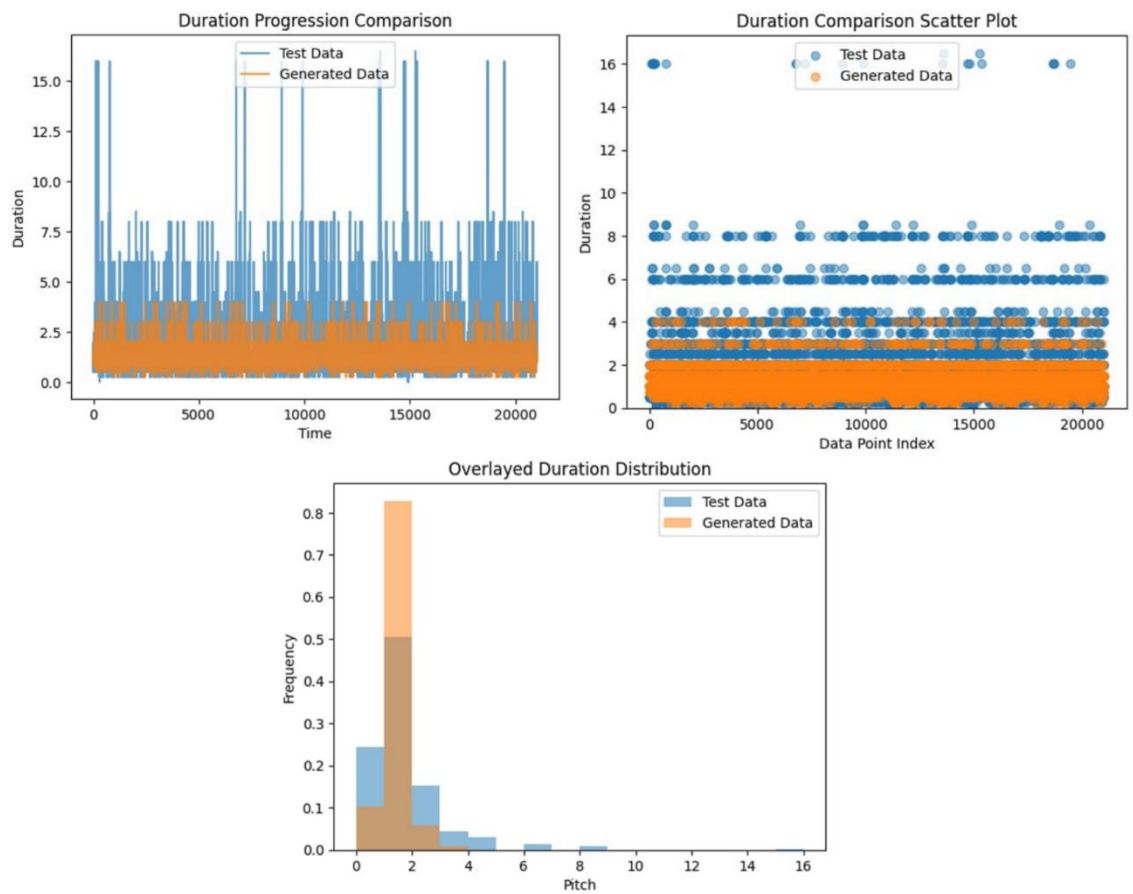


Figure 6.4 various comparison Duration distributions

### 6.1.4 Melody Consistency

In context of analyzing consistency a well-known measure for analyze is counting number of same pitch occurrence, as it can be seen in figure below in two graph could well predict same occurrence of notes, it will lead to a controlled melody following a progression while no chord progression is applied on final output, results here is clearly shows model could capture the lyrics and despite unsuccessful duration prediction it was well succeeded in pitch prediction.

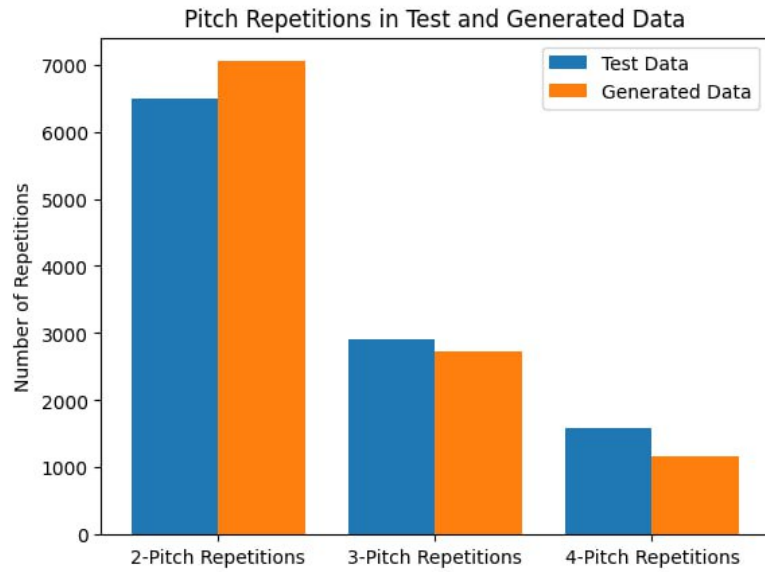


Figure 6.5 pitch repetition in test and generated data

Following research [52] objective metrics to measure similarity of generated data and ground truth melodies are PD (pitch distribution) and DD (duration distribution) to compare with a Transformer Baseline (TB) and two other advanced transformer models.

	PD (%)	DD (%)
<b>TB</b>	24.41	47.04
<b>SONGMASS[52]</b>	34.32	47.88
<b>TELEMELODY[22]</b>	<b>43.86</b>	52.06
<b>MY MODEL</b>	40.58	<b>52.31</b>

Table 6-1 Objective evaluation

MODULE	ACCURACY
<b>PITCH</b>	40.58
<b>DURATION</b>	52.31
<b>REST</b>	70.88

Table 6-2 Model accuracy

## 6.2 Qualitive analysis

In this section qualitative aspects of the melody generation model will be analysed. Unlike the quantitative measures that focus on technical accuracy, the qualitative evaluation focus on the subjective metrics and emotional impact of the generated melodies. This approach involves assessing the melodies based on factors that effects on human listeners, by following study [52] such as harmony, rhythm, structure, and overall quality.

To gain diverse feedback, a group of 10 participants were selected. Each participant was presented with two sets of melodies:

- **Melodies with Lyrics (MWL):** Five melodies paired with lyrics to assess how well the melody complements lyrical content.

- **Melodies without Lyrics (MOL):** Five melodies without any accompanying lyrics to evaluate the standalone quality of the melody.

The reason for this dual approach is to distinguish between the model's ability to generate melodies that are harmonious with lyrics and its ability to create appealing melodies in isolation. Participants rated each melody on a scale from 1 (Poor) to 5 (Perfect). To calculate a comprehensive score, I used the following weighted average formula, where the melodies with lyrics were given double the weight due to the primary focus of our model on lyric-melody synchronization:

$$Total\ Score = \frac{MWL\ score \times 2 + MOL\ Score}{3}$$

MWL and MOL score are obtained from the test participants results to four following questions and rating from 1 to 5:

	DESCRIPTION	FORMULA
<b>HARMONY</b>	How harmonical melody is?	$Harmony\ score\ MWL = \frac{\sum_{i=1}^n Harmony\ score\ i, MWL}{n}$ $Harmony\ score\ MOL = \frac{\sum_{i=1}^n Harmony\ score\ i, MOL}{n}$
<b>RHYTHM</b>	How well lyrics are aligned with lyrics	$Rhythm\ score\ MWL = \frac{\sum_{i=1}^n Rhythm\ score\ i, MWL}{n}$

<b>STRUCTURE</b>	How notes repetitions are impressing	$\text{Structure score MWL} = \frac{\sum_{i=1}^n \text{Structure score } i, \text{MWL}}{n}$ $\text{Structure score MOL} = \frac{\sum_{i=1}^n \text{Structure score } i, \text{MOL}}{n}$
<b>QUALITY</b>	Overall quality	$\text{Quality score MWL} = \frac{\sum_{i=1}^n \text{Quality score } i, \text{MWL}}{n}$ $\text{Quality score MOL} = \frac{\sum_{i=1}^n \text{Quality score } i, \text{MOL}}{n}$

Table 6-3 Subjective metrics

While for Rhythm only MWL score is counted. For the variability in human judgment, a 95% Confidence Interval (CI) for each average score calculated, providing a range within which expected the true average score to lie using following equations:

**Sample Mean ( $\bar{x}$ ):**  $\bar{x}$  = Total score

**Sample Standard Deviation (s):**  $S = \sqrt{\frac{\sum (xi - \bar{x})^2}{n-1}}$

**Standard Error of the Mean (SEM):**  $SEM = \frac{s}{\sqrt{n}}$

CI:  $CI = \bar{x} \pm (t \times SEM)$

Where t is 1.96 for our data

	Harmony	Rhythm	Structure	Quality
BaseLine	2.0 ( $\pm 0.18$ )	2.1 ( $\pm 0.19$ )	1.8 ( $\pm 0.16$ )	1.8 ( $\pm 0.16$ )
SongMass[52]	2.4 ( $\pm 0.20$ )	2.3 ( $\pm 0.20$ )	2.3 ( $\pm 0.20$ )	2.2 ( $\pm 0.19$ )
TeleMelody[22]	<b>3.2 (<math>\pm 0.24</math>)</b>	<b>3.4 (<math>\pm 0.19</math>)</b>	<b>3.3 (<math>\pm 0.21</math>)</b>	<b>3.3 (<math>\pm 0.20</math>)</b>
MyModel	2.6 ( $\pm 0.39$ )	2.8 ( $\pm 0.72$ )	2.7 ( $\pm 0.28$ )	2.6 ( $\pm 0.19$ )

Table 6-4 subjective evaluation (with 95% confidence interval)

Below are some melodies generated using my model:

$\text{♩} = 120$

6

$\text{♩} = 120$

10

$\text{♩} = 120$

6

$\text{♩} = 120$

Figure 6.6 generated melodies

## 7 Conclusion

The testing phase of the model demonstrated its ability to accurately interpret the content of lyrics, predicting pitches and, in some cases, the duration of notes. While the generated musical data remained logically consistent, it became apparent that further training could enhance the model's accuracy. This suggests that with more extensive training periods, the model's proficiency in creating melodies could be significantly improved.

The results of this study indicate that this AI method is a viable solution for melody generation. It shows promise in understanding and translating the essence of lyrics into musical notes. However, the process of training such a model is not without its challenges. It requires substantial computing resources and advanced GPUs, making it a complex task in a standard computing environment.

A noteworthy contribution of this research is the introduction of a dataset that pairs melodies with corresponding lyrics, even at the sentence level. Such comprehensive and varied data is rare, and this dataset is expected to be an asset for future research in this field.

In response to the central question of whether AI can replace humans in song composition, the answer leans towards yes, but with specific conditions. The success of AI in this domain hinges on the availability of ample resources, more advanced data access, and further refinements in AI technology. This research opens the door to future explorations in AI music composition, underscoring the potential for machines to create art that resonates with human emotions and sensibilities.

Also generated melodies using this model are accessible throw this google drive[50]

## Acknowledgements

I am immensely grateful to my supervisor, Grad-Gyenge László György, for his exceptional guidance and support throughout this project and my academic journey in both the training and software laboratories over the past semesters. His profound knowledge, insightful mentorship, and patient guidance have been pivotal in my growth and understanding of complex concepts in our field. His ability to provide constructive feedback, coupled with challenging and inspiring assignments, has greatly honed my skills, and fostered a deep sense of curiosity. Mr. László György's dedication to creating a supportive and stimulating learning environment has not only aided me in successfully completing this project but also in preparing for future academic and professional pursuits. His commitment to excellence and unwavering support have left an indelible mark on my educational journey, for which I am eternally thankful.



## References

- [1] Rizo, D. (2006). *A Pattern Recognition Approach for Melody Track Selection in MIDI Files*. <https://www.semanticscholar.org/paper/A-Pattern-Recognition-Approach-for-Melody-Track-in-Rizo-Le%C3%B3n/f8b17b3ebeb0b8ba4e308b931e987e2f3392428b>
- [2] Velusamy, S., Thoshkahna, B., & Ramakrishnan, K. R. (2006, January 1). *A Novel Melody Line Identification Algorithm for Polyphonic MIDI Music*. Lecture Notes in Computer Science. [https://doi.org/10.1007/978-3-540-69429-8\\_25](https://doi.org/10.1007/978-3-540-69429-8_25)
- [3] Wen, R. (2019). *Music Main Melody Extraction by An Interval Pattern Recognition Algorithm*. <https://www.semanticscholar.org/paper/Music-Main-Melody-Extraction-by-An-Interval-Pattern-Wen-Chen/e9f8b5830e5174b2eef087a8b8a67082a2b7dad3>
- [4] Uitdenboger, A. L., & Zobel, J. (1998, January 1). *Manipulation of music for melody matching*. <https://doi.org/10.1145/290747.290776>
- [5] Martín, R. B., Mollineda, R. A., & García, V. (2009, January 1). *Melodic Track Identification in MIDI Files Considering the Imbalanced Context*. Lecture Notes in Computer Science. [https://doi.org/10.1007/978-3-642-02172-5\\_63](https://doi.org/10.1007/978-3-642-02172-5_63)
- [6] Raposo, F., De Matos, D. M., & Ribeiro, R. (2021, May 29). *Assessing kinetic meaning of music and dance via deep cross-modal retrieval*. Neural Computing and Applications. <https://doi.org/10.1007/s00521-021-06090-8>
- [7] Zhao, J., Taniar, D., Adhinugraha, K. M., Baskaran, V. M., & Wong, K. (2023, August 16). *Multi-mmlg: a novel framework of extracting multiple main melodies from MIDI files*. Neural Computing and Applications. <https://doi.org/10.1007/s00521-023-08924-z>
- [8] Huron, D. (2001, January 1). *Tone and Voice: A Derivation of the Rules of Voice-Leading from Perceptual Principles*. Music Perception. <https://doi.org/10.1525/mp.2001.19.1.1>
- [9] Özcan, G., Işıkan, C., & Alpoçak, A. (2006, January 5). *Melody Extraction on MIDI Music Files*. <https://doi.org/10.1109/ism.2005.77>
- [10] Simonetta, F., Cancino-Chacón, C., Ntalampiras, S., & Widmer, G. (2019, June 24). *A Convolutional Approach to Melody Line Identification in Symbolic Scores*. arXiv (Cornell University). <https://doi.org/10.5281/zenodo.3527965>
- [11] Cambouropoulos, E. (2008, September 1). *Voice And Stream: Perceptual And Computational Modeling Of Voice Separation*. Music Perception. <https://doi.org/10.1525/mp.2008.26.1.75>
- [12] Friberg, A. (2009). *Recognition of the Main Melody in a Polyphonic Symbolic Score using Perceptual Knowledge*. DIVA. <https://kth.diva-portal.org/smash/record.jsf?pid=diva2%3A337274&dswid=375>

- [13] *A Novel Extraction Method for Melodic Features from MIDI Files Based on Probabilistic Graphical Models*. (2018, August 1). IEEE Conference Publication | IEEE Xplore. <https://ieeexplore.ieee.org/document/8597928>
- [14] “Normal Distribution.” *Wikipedia*, 13 Dec. 2023, [en.wikipedia.org/wiki/Normal\\_distribution](https://en.wikipedia.org/wiki/Normal_distribution).
- [15] Fernández, J., & Vico, F. J. (2013, November 17). *AI Methods in Algorithmic Composition: A Comprehensive Survey*. Journal of Artificial Intelligence Research. <https://doi.org/10.1613/jair.3908>
- [16] Anders, T., & Miranda, E. R. (2011, October 1). *Constraint programming systems for modeling music theories and composition*. ACM Computing Surveys. <https://doi.org/10.1145/1978802.1978809>
- [17] Dai, S. (2021, September 2). Controllable deep melody generation via hierarchical music structure representation. arXiv.org. <https://arxiv.org/abs/2109.00663>
- [18] Hornel, D. (1998). *Learning musical structure and style with neural networks*. <https://www.semanticscholar.org/paper/Learning-musical-structure-and-style-with-neural-Hornel-Menzel/d6952abf61b2b553253e9265c9b96d9af2f7523b>
- [19] Zhou, Y. (2018, December 18). BandNet: A Neural Network-based, Multi-Instrument Beatles-Style MIDI Music Composition Machine. arXiv.org. <https://arxiv.org/abs/1812.07126>
- [20] Wang, Z. (2018, December 28). A Framework for Automated Pop-song Melody Generation with Piano Accompaniment Arrangement. arXiv.org.
- [21] Watanabe, K., Matsubayashi, Y., Fukayama, S., Goto, M., Inui, K., & Nakano, T. (2018, January 1). *A Melody-Conditioned Lyrics Language Model*. <https://doi.org/10.18653/v1/n18-1015>
- [22] Ju, Z. (2021, September 20). TeleMelody: Lyric-to-Melody Generation with a Template-Based Two-Stage Method. arXiv.org. <https://arxiv.org/abs/2109.09617>
- [23] *Phoneme*. (2023, December 6). Wikipedia. <https://en.wikipedia.org/wiki/Phoneme>
- [24] Lv, A. (2022, August 11). *Re-creation of Creations: A New Paradigm for Lyric-to-Melody Generation*. arXiv.org. <https://arxiv.org/abs/2208.05697>
- [25] Zhang, C., Chang, L., Wu, S., Tan, X., Qin, T., Liu, T. Y., & Zhang, K. (2022, October 10). *ReLyMe*. Proceedings of the 30th ACM International Conference on Multimedia. <https://doi.org/10.1145/3503161.3548357>
- [26] Lu, P. (2022, August 30). MeloForm: Generating Melody with Musical Form based on Expert Systems and Neural Networks. arXiv.org. <https://arxiv.org/abs/2208.14345>
- [27] [https://en.wikipedia.org/wiki/Music\\_theory](https://en.wikipedia.org/wiki/Music_theory)

- [28] Anssi Klapuri, "Introduction to Music Transcription", in *Signal Processing Methods for Music Transcription*, edited by Anssi Klapuri and Manuel Davy, 1–20 (New York: Springer, 2006): p. 8. ISBN 978-0-387-30667-4.
- [29] B. (2022, January 28). *Note Lengths*. Music Theory Academy.  
<https://www.musictheoryacademy.com/how-to-read-sheet-music/note-lengths>
- [30] *5 Basic Elements of Music*. (2013, September 18). Play Notes.  
<https://playnotes.wordpress.com/2013/09/17/5-basic-elements-of-music>
- [31] Yu, Y., Srivastava, A., & Canales, S. (2021, February 28). Conditional LSTM-GAN for Melody Generation from Lyrics. *ACM Transactions on Multimedia Computing, Communications, and Applications*. <https://doi.org/10.1145/3424116>
- [32] *Circle of fifths*. (2023, December 4). Wikipedia.  
[https://en.wikipedia.org/wiki/Circle\\_of\\_fifths](https://en.wikipedia.org/wiki/Circle_of_fifths)
- [33] Nichols, E. P., Morris, D., Basu, S., & Raphael, C. (2009, January 1). *Relationships Between Lyrics and Melody in Popular Music*. ResearchGate.  
[https://www.researchgate.net/publication/220723643\\_Relationships\\_Between\\_Lyrics\\_and\\_Melody\\_in\\_Popular\\_Music](https://www.researchgate.net/publication/220723643_Relationships_Between_Lyrics_and_Melody_in_Popular_Music)
- [34] Clark, J. (1990). *An Introduction to Phonetics and Phonology*.  
<https://www.semanticscholar.org/paper/An-Introduction-to-Phonetics-and-Phonology-Clark-Yallop/d4828e5d38aa8ce7b1831b1b02df8a923acc17ce>
- [35] *Natural language processing*. (2023, November 28). Wikipedia.  
[https://en.wikipedia.org/wiki/Natural\\_language\\_processing](https://en.wikipedia.org/wiki/Natural_language_processing)
- [36] *Natural Language Toolkit*. (2023, May 15). Wikipedia.  
[https://en.wikipedia.org/wiki/Natural\\_Language\\_Toolkit](https://en.wikipedia.org/wiki/Natural_Language_Toolkit)
- [37] *Tokenizer*. (n.d.). [https://huggingface.co/docs/transformers/main\\_classes/tokenizer](https://huggingface.co/docs/transformers/main_classes/tokenizer)
- [38] *Word2vec*. (2023, September 5). Wikipedia.  
<https://en.wikipedia.org/wiki/Word2vec>
- [39] *Transformer (machine learning model)*. (2023, December 14). Wikipedia.  
[https://en.wikipedia.org/wiki/Transformer\\_\(machine\\_learning\\_model\)](https://en.wikipedia.org/wiki/Transformer_(machine_learning_model))
- [40] Vaswani, A. (2017, June 12). *Attention Is All You Need*. arXiv.org.  
<https://arxiv.org/abs/1706.03762>
- [41] *The Lakh MIDI Dataset v0.1*. (n.d.). <https://colinraffel.com/projects/lmd/>
- [42] Colin Raffel. *"Learning-Based Methods for Comparing Sequences, with Applications to Audio-to-MIDI Alignment and Matching"*. PhD Thesis, 2016
- [43] Thierry Bertin-Mahieux, Daniel P. W. Ellis, Brian Whitman, and Paul Lamere. "The Million Song Dataset". In *Proceedings of the 12th International Society for Music Information Retrieval Conference*, pages 591–596, 2011

- [44] Dong, H. W. (2017, September 19). MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment. arXiv.org. <https://arxiv.org/abs/1709.06298>
- [45] Papers with Code - MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment. (2017, September 19). <https://paperswithcode.com/paper/musegan-multi-track-sequential-generative>
- [46] Azad, M. I. (2023, June 8). Sequence-to-Sequence Model with Transformer-based Attention Mechanism and Temporal Pooling for Non-Intrusive Load Monitoring. arXiv.org. <https://arxiv.org/abs/2306.05012>
- [47] Vaswani, A. (2017, June 12). *Attention Is All You Need*. arXiv.org. <https://arxiv.org/abs/1706.03762>
- [48] *Feedforward neural network*. (2023, October 3). Wikipedia. [https://en.wikipedia.org/wiki/Feedforward\\_neural\\_network](https://en.wikipedia.org/wiki/Feedforward_neural_network)
- [49] *Mean squared error*. (2023, December 6). Wikipedia. [https://en.wikipedia.org/wiki/Mean\\_squared\\_error](https://en.wikipedia.org/wiki/Mean_squared_error)
- [50] *The CMU Pronouncing Dictionary*. (n.d.). <http://www.speech.cs.cmu.edu/cgi-bin/cmudict>
- [51] [https://drive.google.com/drive/folders/1hTPwUYPr5W5NfPJUdYVObev6Vp7GVoC4?usp=drive\\_link](https://drive.google.com/drive/folders/1hTPwUYPr5W5NfPJUdYVObev6Vp7GVoC4?usp=drive_link)
- [52] Sheng, Zhonghao. "SongMASS: Automatic Song Writing With Pre-training and Alignment Constraint." *arXiv.org*, 9 Dec. 2020, [arxiv.org/abs/2012.05168](https://arxiv.org/abs/2012.05168).