Title of your thesis

Your Name

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Your Department

Your Advisor, Chair
First Committee
Second Committee
Third Committee
Last Committee

December 4, 2020
Blacksburg, Virginia

Keywords: Some Keywords, Subject matter, etc.

Title of your thesis

Your Name

ABSTRACT

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Dedicated to Virginia Tech.

Acknowledgments

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Contents

List of Figures

List of Tables

Todo list

Chapter 1

Introduction

In 1952 L.A. Hiller and L.M Issacson ushered forth a new era of the study of both music and computer science when they introduced the Illiac Suite – the first composition that was created solely by a computer [32]. What we'll refer to broadly as Music Information Retrieval (MIR)[1] research has continued to see impressive advancements since the introduction of the Illiac Suite in several different domains, including musical composition[3], instrument and sound synthesis[11], and musical analysis[35]. Much of the challenge in MIR research is to bridge the gap between the highly subjective nature of the musical experience paired with the strong hierarchical and mathematical patterns that are present from a quantitative perspective. The most well-known application of MIR research is that of musical recommendation platforms such as Spotify which study the different patterns present in a diverse range of musical ideas to suggest to their listeners music which they may appreciate in the future given their past listening habits.

Widmer[35] suggests that there are a deeper set of problems that the MIR community should focus on which are by their nature, more fundamental to understanding the nature of music itself. One such problem is that of understanding what constitutes *expression* in musical performance. Current automatic performance generation systems (typically bundled with musical notation software) render deterministic and uninteresting performances which don't

[1] Widmer [35] points out that MIR itself does not encompass the entire scope of computer music research, but that it is a good proxy to use when referring the field as a whole. We will operate under the same assumption

contain the ”human” element; that is, they do not use the different components of musical performance such as variations in timing, dynamics and articulation, to ”express” different ideas or emotions. Each performance of a musical composition inherently carries with it some interpretation of the composition which is communicated through musical expression. Our work is a further continuation of the computational modeling of expressive musical performance (EMP) in the context of automatic performance generation.

Typical EMP generation systems use Machine Learning (ML) to build generation systems which are trained on existing data sets comprised of actual human performance. The most recent models are either probabilistic (usually using Hidden Markov Models) in nature, or based on artificial neural networks (ANN) which is a family of ML models that have led to the rapid increase of Artificial Intelligence (AI) systems in many areas, including computer vision, natural language processing (NLP), and speech and audio processing[15][2]. State of the art models are based on Recurrent Neural Networks (RNN) and their common adaptation as a Long Short Term Memory (LSTM) network which are designed to model sequential data, such as music. A relatively new model in sequential Deep Learning (DL), the Transformer, has led to impressive advances over RNN based models in NLP and other fields. We apply the Transformer in EMP generation, which to our knowledge has never been done, using an existing end to end state of the art EMP generation system.

We evaluated our model quantitatively through the current standard evaluation metric and it performs worse than an existing RNN based system. However, a qualitative evaluation through our personal listening revealed a disconnect between the performance according to the quantitative metric and performance according to actually listening to the performances. We ran further experiments deviating from the standard method to identify why this might be

[2]The use of neural networks in ML is commonly referred to as ”Deep Learning” because of the many connected layers that usually comprise the networks. The term ”deep” is used to describe the long path which information must follow to propogate through the large network. This is in contrast to other ML models which usually do not have that depth

the case, and provide some insights about possible problems with the quantitative evaluation and intuition for how to create better evaluation methods. We also give an error analysis of our own model and notice that the decrease in performance may not be with the underlying Transformer mechanisms but with our network architecture from a higher level.

We also bring to light some of the philosophical conundrums with studying EMP generation at a computational level. Because the "quality" of a musical experience is so subjective, creating the right incentives for a computer model to generate novel performances is not trivial. This is related to our discovery of possible flaws in our model evaluation. In agreement with other authors [35], we advocate for further research in the area to draw just as much from music at the human psychological level as the mathematical and statistical.

Chapter 2

Background: Expressive Performance

There are two major research components upon which this project is based. The first is the problem domain of expressive musical performance, and the second is the ML modeling domain of Transformers. In this chapter, we will introduce and discuss EMP. In chapter 3 we'll do the same for the Transformer.

Expressive musical performance is a small subset of Music Information Retrieval research, which can be broadly categorized into two separate tasks: the first is developing computational methods for musical analysis, and the second is developing computational methods for music generation. We are interested in the latter, although it is worthwhile to note that there is a large overlap between the areas[1]. In order to study how musical performance generation (and more particularly *expressive* musical performance generation) models work, it is necessary to gain a proper understanding of the entire computational musical generation process as a whole. Ji et al. [24] break the process down into 3 different components, with 4 different roles or agents that interact with that process. Figure 2.1 shows each step in the process as well as the agents that participate

Richard: Try to get permission to reproduce the image in the paper

.

[1]Creating a performance generation system is useful for performance analysis as long as the generation system is interpretable. The same can be said in reverse. Analysis can provide insight to generation, and generation can provide insight to analysis

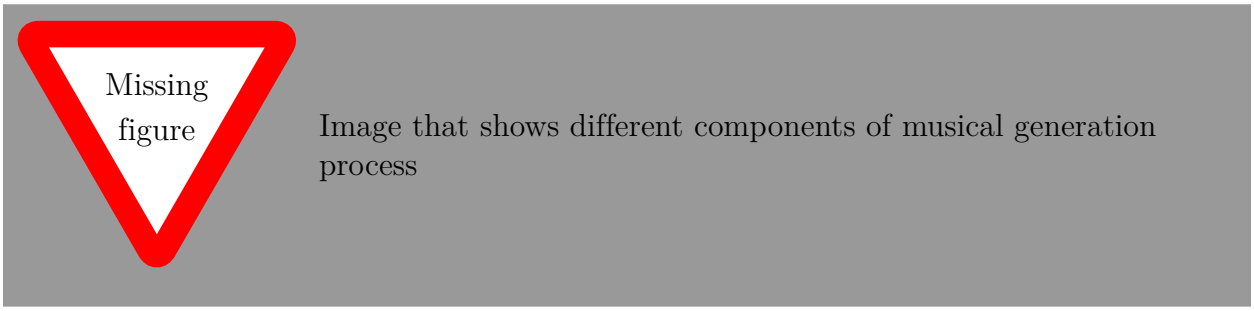

Figure 2.1: The first step of musical generation is composition, shown as a score in the figure. The second is performance, which is our area of interest. The third is the production of sound. Each different agent: composer, performer, instrument, and listener, can be thought of as a separate computational model in the generation process

An EMP generation model is analogous to the performer as shown in 2.1, who takes as input a musical composition and produces as output a performance. It is the phenomena of musical expression that makes the performance generation process interesting. Musical expression can be thought of as the performers' interpretation of a composition codified into different performance parameters that are intended to contribute the quality of a musical experience [2]

To provide a more detailed explanation of expression in musical performance, it is necessary to clearly define the first two components of the generation process - namely, scores and performances. We provide descriptions of both components at both a general and mathematical level. At the mathematical level we use the terminology of a **feature**, which is commonly used to describe the numerical values and data structure which are used as the input and output of a ML model. We refer the reader to appendix A.1 which provides some basic musical terminology and concepts that will be useful for understanding our definitions[3]. Due

[2]Because the quality of a musical experience is highly subjective, there is no definition of what makes for a "correct" interpretation of a given composition [4]. The subjective nature of EMP generation makes it a difficult problem to understand from a computational perspective and is related to our discussion of evaluation methods given in section 2.3.2.

[3]Most of the appendix material may seem elementary to those who already have a background in music or musical notation. However, we feel that is necessary to include if for no other reason than to provide a clear definition for our descriptions both in general and at detailed mathematical level

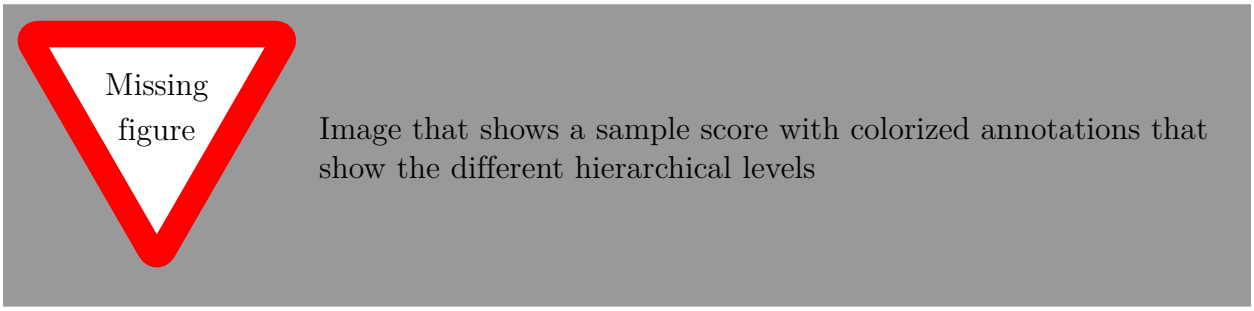

Figure 2.2: Caption will be dependent on the image.

to the constraint of our data we focus only on western classical piano music.

Richard add reference

2.1 Scores

A musical score is a symbolic representation of a musical composition. The symbolic notation used to create musical scores can be thought of as a language used to express musical ideas and information. It presents this information in a hierarchical structure with different levels of musical detail at each level. Figure 2.2 shows a sample score and the different hierarchical levels of information that it contains

The lowest level contains information about the pitch and timing of every single note, as well as optional information about how the note should be played. This can include information specific to instruments such as the bow direction of a violin, but for our purposes (dealing only with piano) we will consider this to be the articulation of each note, usually indicated by ***legato*** or ***staccato***

Richard: Make sure to have some background information on articulation in the appendix

The middle level contains information related to certain substructures within the musical composition, which are usually expressed within a grouping of notes or measures. The most

common score annotations at this level are dynamic markings which indicate whether to play a grouping of notes as ***f*** (loud), ***p*** (soft), or as a ***crescendo*** or ***decrescendo*** (gradually increase or decrease the volume). Although dynamic markings are the most common at this level, it is also possible to see score markings for all other musical features, such as local tempo or articulation of a certain substructure. Perhaps the most important score marking at this level is that of a phrase, which is a marking that indicates that a group of notes should be interpreted as belonging to a singular musical idea and that each note should fit within the context of the phrase as a whole. A phrase can be expressed through all of the different aforementioned musical features, including the tempo, timing, dynamics, and articulation of the notes.

The highest level contains meta-information that relates to the entire composition as a whole. This information typically includes the key signature and time signature, as well as the global tempo for the entire piece, most commonly represented as BPM.

2.1.1 Score Features

Richard: This entire section needs work. Include some detailed information but refer reader to chacon's thesis for a full breakdown. Possibly mention features from virtuosoNet

.

There are some score features which are required for EMP models, which include the musical features at the lowest level of a score as explained in section 2.1. These are pitch and timing, and the duration of the notes. Mid-level features include concepts at the local level and have some music theoretic concepts, such as downbeat information of a given measure according to the time signature, or the tonality of a chord (tonic, dominant, etc). High-level features

represent advanced music theoritic concepts that are more global to the entire piece, including abstract properties of the piece such as the emotion the piece should convey and how different sections of the piece relate to each to tell a complete story [10].

Both the mid-level and high-level features are not necessarily required for every EMP model as the lower-level features are, and are not consistent across all EMP models. It still remains an open question as to which features should be extracted from the data that the model can learn from. The lack of consistency in these features is one of the reasons that evaluation of EMP generation models is so difficult, as explained in section 2.3.2.

2.2 Peformance

An expressive musical performance contains most of the same musical information as does a score, but with one key difference; that is, that an expressive performance will deviate (or interpret) from the exact information that is presented in the score. For example, although a score may indicate a tempo of 120 BPM, it is highly unlikely that a given performer will perfectly adhere to this tempo throughout the entirety of the piece. This is even more apparent if the score indicates a change in tempo somewhere in the composition. If a score indicates that the performance should speed up over a series of notes, there is no telling at what rate the tempo should increase. Some performers may choose to speed up at a fast rate and over a short period of time. Others may choose to increase the tempo at a slow rate and over a longer period of time. A single accelerando (a score indication to pick up the tempo) can result in either of these outcomes.

As mentioned a performance contains most of the same musical information related to a score, which include pitch, tempo, timing and articulation. Each of these expressive parameters will be measurable and absolute, whereas the score markings of these features can be viewed

more as a suggestion than a rule. There a few additional components of performance that are not ncessarily indicated in scores but are relevant which in understanding performance. The first we will refer to as deviation which is heavily related to timing. It is typically represented as a numerical number which represents how far off the timing of a particular note deviates from it's "correct" position in the score. These micro-timing deviations present in musical performances are an essential part of expression. Without them, indicating that each note onset and offset is exactly in line with its marking in the score, performances sound robotic and mundane

Richard: add a reference, graphic, and sample performance

.

The other important feature of performance that is not always present in a score applies specifically to the piano, and is the presence of a piano pedal. There are several different types of piano pedals, but the most common are the sustain pedal, which prolongs the duration of every note of the piano when activated, and the soft pedal which softens the sound of the entire piano. Although the effects of these pedals are directly related to the articulation and dynamics of the performance, their presence (or lack of) can be seen as a crucial component of piano performance. It is common for the sustain pedal to see active use in almost all modern piano performance, even when there doesn't exist any score marking indicating it's use.

2.2.1 Performance Features

Richard: Similarly to score features section, provide more detailed information about some of the math behind the features. Cite other resources where necessary

For western classical solo piano music, performance features are relatively simple compared to

the score features as well as to other instruments. Most EMP models use the different aspects of a piano performance as explained in section 2.2 for their data features, including the pitch, tempo, timing (or timing deviation), articulation, and pedal. Although at an abstract level the features are the same, there are different numerical methods used to describe each of the different aspects. These are presented in

Richard: Add reference to relevant work section that goes over the different elements. This information may belong better here and then referenced in the relevant work section

.

2.3 Data

The data required for EMP generation includes some digital form of representation of a score as well as a corresponding performance. Scores are typically given in the form of MusicXML, which is a text-based representation of a score. Performances could be directly be rendered as audio which is the process used by human performers with the use of an acoustic instrument. Instead of audio however, an intermediate data form, MIDI, is used to represent the performance. This better aligns with the generation process outlined in 2.1. In the full generation process, a separate model would be used to take the performance data in MIDI and synthesize that into raw audio which would be presented to the listener. Both data formats contain all of the required information to represent all of the musical components of both a score and a performance, including pitch, tempo, timing, articulation, deviation, and pedal. See appendix A.2 for more information on both MusicXML and MIDI.

To build an EMP generation model, it is necessary to run both the score and performance through a data alignment process in which every note of the performance is mapped to it's

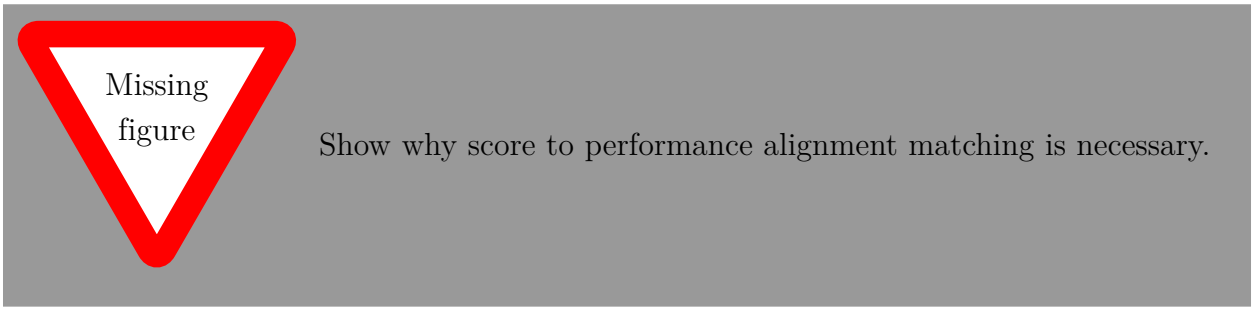

Figure 2.3: Two performances of the same score can vary wildly in their tempo and timing. This makes it necessary to have a score to performance alignment for every performance.

corresponding position in the score. Given the highly dynamic nature of musical performance, it is a non-trivial task to run this alignment process for a set of scores and performances, especially if the task is performed by manual human annotation. There exist methods for both manual and automatic alignment. Due to the time-consuming nature of manual alignment and the need for large data sets to build higher quality models, automatic alignment algorithms are an active area of research.

2.3.1 Existing Data Sets

Richard: Rework this section to fit with new paper outline. Need to forward reference models instead of back reference

One of the problems facing EMP and MIR in general is the lack of high quality and high scale datasets[4]. This is in large part due to the fact that scope of possible data to collect related to music data is large, compared to other domains. As has already been discussed, there are different stages in the musical process, and each of them contain different possibilities for the representation of music. For example, composition can contain largely the same amount of information in at least three forms. The first and most common is the symbollic representation in the form of a data format like MusicXML. A musical performance also con-

tains within it information about the composition itself, and performances can represented in an intermediate format such as MIDI, or in the form of raw audio. The same can be said for other fields such as NLP, which deals mostly with textual data, and Speech Processing, which deals mostly with language in the form of spoken word. However, the two fields are seen as distinct from each other and each come with more standardization in both research methods and data formats. Musical data and information has not seen the same rigour in the literature.

Another inherent problem with getting high-quality musical datasets is that most of the readily available musical data comes in the form of audio, which is much more difficult to process than symbolic (MusicXML) or intermediate (MIDI) forms given that it contains large amounts of noise and does not necessarily compress musical information. In contrast, NLP and Computer Vision directly deal with text and image data respectively, which are both readily available at a large scale due to the internet.

There are normally 3 required components for a EMP dataset.

1. Scores (usually in the form of MusicXML)
2. Performances (usually in the form of MIDI)
3. Metadata about the matching alignment between the score and performance.

Score data is usually gathered by finding readily available MusicXML files from open source software projects which contain music that is in the public domain (which all western classical music is) [4], or by using Optical Music Recognition (OMR) to auotomatically scan paper sheet music into a digital form followed by manual corrections where needed. Because the relevant performance features are difficult to extract from raw audio, performance data usually comes

[4] MuseScore is the most common. Also see the International Music Score Library Project

in the form of MIDI. To gather MIDI data of professional performance, it is necessary for the performers to play on a computer-controlled piano which can record performances in MIDI form, as well as automatically play back recorded performances which allow the complete reproducibility of any existing performance. Both the Yamaha Disklavier [5] and the older Bosendorfer CEUS system have this capability.

There is no standardized method for score-to-performance alignment methods and data representations. Each dataset presents in own alignment method as well as the metadata that represents the alignment.

To provide context for the progression of data used in EMP generation, we'll start by touching an older dataset used in older EMP research, the Magaloff Corpus, and then describing a much larger scale dataset, the Piano-e-competition, which has recently been adapted for use in EMP generation. A full overview of datasets for EMP generation can be found in [4]

Magaloff Corpus

Nikita Magaloff was a Russian pianist known for his performance cycles of Chopin's entire works for the solo piano. In one of his final cycles of performances recorder in 1989, he played on a Bosendorfer SE computer-controlled piano. The Magaloff Corpus [12] presented the recorded performances were converted to the standard MIDI format[10], thus making available full performance data of all of Chopins compositions for solo piano. Score data was obtained using OMR with manual corrections where needed. The alignment method presented in [16] was used to produce the note-matching annotations, along with manual correction. The dataset contains over 10 hours of playing, 150 compositions, and over 320,000 performed notes. The corpus however, is not publicly available, and has only been used in research by Flossmann et al. [12] and colleagues [10].

[5] https://usa.yamaha.com/products/musical_instruments/pianos/disklavier/index.html

Piano-e-competition

As has been discussed, there is a large push in modern MIR to produce high-quality large datasets. At the heart of this research in MIR is the Piano-e-competition. Started in 2002, it is an international piano competition which attracts some of the promising up and coming musicians at both the senior and junior level [1]. Every performance from the competition is played on a Yamaha Disklavier and is recorded in both MIDI and audio. Much of the research in MIR and music generation uses this dataset due to its size and availability. As such, there exist several different adaptations of the original data which are specific to certain research purposes.

The first of these is the MAESTRO (Midi and Audio Edited for Synchronous TRacks and Organization) dataset. Hawthorne et al. [18] introduce the MAESTRO dataset, which presents both MIDI and audio data from the Piano-e-competition in a canonical and easily accessible form. The dataset was first used to build a full musical analysis and generation process framework named wav2midi2wave. This framework includes a musical transcription process [17] from raw audio to midi (wav2midi), a direct musical composition and performance generation model [19] [6] (can be seen as the midi or midi2midi part of the wav2midi2wav framework), and a synthesis model that takes MIDI and generates raw audio[28] (midi2wav). The MAESTRO dataset is the most commonly used form of the Piano-e-competition data.

The Piano-e-competition also forms the basis for a data collection, which we will refer to as the KAIST dataset[7]. The Piano-e-competition dataset itself does not provide any score data about any of the compositions used in performance. The KAIST dataset was created specifically for an EMP generation system, and therefore needs score data for every perfor-

[6] This model directly generates MIDI files without using scores. It simultaneously generates a composition and performance. This direct generation is a merging of the two separate tasks into one as shown in figure 2.1

[7] taking the name from the KAIST Graduate School of Technology, which the researchers who created the dataset work for

mance recorded in MIDI. This score data was collected by Jeong et al. [21] by downloading MusicXML files online, mostly from MuseScore. On top of gathering the score data for all performances in the Piano-e-competition, Jeong et al. [21] also run the automatic score-to-performance alignment algorithm of Nakamura et al. [26] to provide metadata about the alignment between each score and performance. Automatic score-to-performance alignment is error-prone, especially in the case of performance mistakes[8] As a result, some performance notes are not aligned to those in a score. Due to the possibility for error in automatic alignment, Jeong et al. [21] also add additional manual and heuristic corrections to the alignment where needed.

The difference between the KAIST dataset and the Magaloff corpus contains is that the KAIST dataset contains multiple performances for the same score. In contrast, the Magaloff Corpus has a 1-1 mapping between a score and performance. The KAIST dataset has 226 scores across 16 different composers, roughly 660,000 score notes, and around 3,500,000 performance notes. The number of matched performance notes is ten times larger than the Magaloff Corpus, and all data is publicly available [9].

The Aligned Scores and Performances (ASAP) dataset [13] is a recent adaptation of both the KAIST dataset and the MAESTRO dataset. It uses the MusicXML files from the KAIST dataset, audio from the MAESTRO dataset, and MIDI files from both sources extracted from the common origin of the Piano-e-competition. It provides additional alignment metadata for both MIDI and audio and more manual correction in the MusicXML score files. Although the purpose of the ASAP dataset is for Automatic Music Transcription (AMT)[10], it is just as equally useful for EMP generation. To our knowledge, it hasn't seen an application in any EMP generation task. Although it is mostly similar to the KAIST dataset, the implications

[8] A mistake in the performance results in a performance note having no correct match with a score note.

[9] The dataset is open sourced at https://github.com/mac-marg-pianist/chopin_cleaned

[10] AMT is the task of transcribing a score from a performance (either in audio or MIDI form). AMT is the "opposite" of EMP. It maps a performance to a score instead of a score to performance

of its extensions are undetermined in EMP research.

2.3.2 Performance Evaluation

Richard: Should probably go in chapter 3

One of the most important components of any computational model performing a task is that of evaluation. Evaluation is used to determine the quality of a model, and serves as a benchmark to compare different models used in the same task. Due to the inherently subjective nature of music and musical performance discussed in **??**, evaluation is notioursly difficult to understand and perform correctly for EMP generation models [4].

Evaluation for computational models, specifically for EMP models, is typically categorized in two ways, quantitative evaluation and qualitatiative evaluation. Quantitative evaluation methods involve using numerical metrics which are computationally generated and deterministic. Qualitiative evaluation methods usually involve some form of human feedback and judgement presented in some standardized statistical measures. The key difference between quantitative and qualitiative is that qualitative methods are not as consisten and much more difficult to reproduce, given the reliance on the subjective feedback of human listeners. Traditionally, quantitative methods are preferred because of their consistency and reliability, In the case of EMP models however, qualitative evaluation methods may be even more important in gaining an understanding of what makes one model better than another. Finding good methods of evaluation is an active area of research in EMP [4].

Quantitative

This method of evaluation is standard for ML models in general. There are a number of different metrics which are used in the evaluation process, all of which are specific to type of

data and problem domain the model fits inside of. We will briefly cover the most common quantiative evaluation method that applies to our data and modeling domain, which is regression .

Richard

Add reference that discusses the feature engineering

The two common metrics used for evaluation and regression are Mean-Squared-Error (MSE) and the Pearson Correlation Coefficient, usually denoted as the R^2 score. MSE is used to measure the difference between a prediction and an actual observed target value, and can be denoted as $MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$, where Y_i is the observed value at time step i, and $\hat{Y}_1$ is the predicted value. R^2 is a probablistic measure of the linear correlation between variables X and Y, and is denoted as $\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$ where cov indicates the covariance and σ indicates the standard deviation. [11]

One of the problems with using quantiative, or "objective" evaluation methods, is that it usually invovles comparing a generated performance $\hat{Y}$ with a human performance Y. Given that no performance (or interpretation) of a can objectively been seen as better than another, this method of evaluation is also biasing the quality of a model towards some subjective view of the "correct" interpretation. Of course, a "correct" interpretation doesn't exist, which is what makes evalution methods for this particularly problem difficult.

Richard: Add reference to other works using either MSE or r2

Qualitative

Richard: Need to conduct more research before I can write this section. Haven't done so because I won't be performing a qualitative evaluation myself in the paper. However it is still worth mentioning

[11] See wikipedia for more information on MSE, covariance, standard deviation, and the correlation coefficient

Chapter 3

Background: Sequential Modeling

This project's motivation comes from the context set by the current state of the art of EMP generation and ML in general. In the following chapter, we provide some of this context and its relevance to our problem. Current state of the art models in performance generation use Recurrent Neural Networks as their foundation. Recent developments in neural sequence modeling move entirely away from RNNs and toward a new family of ANN architectures, the Transformer. Because Transformers have not seen application in EMP generation models, they are the focus of our work.

3.1 Sequential Data

One fundamental aspect of modern machine learning is modeling sequential data. Sequential data consists of individual data points with a relationship to each other according to some specific order and position in time. A simple example of sequential data is the weather, which follows predictable patterns according to the time of year. Musical data is also fundamentally sequential[35] given that we experience music as events that happen in time, and the relationship of such events according to their position in time is paramount to the phenomena of musical experience. Language and speech exhibit this same property, and the research in natural language processing drives much of the research advance in neural sequential data modeling.

Sequential data modeling is typically categorized into different tasks. Perhaps the most common task is sequence classification, which seeks to assign some sequence of data points to a particular class of data. In this case, input data is defined as $X = \{x_1, x_2, x_3, ..., x_n\}$ where $x_i \in \mathbb{R}^m$. Output data is a single value $y \in L$ where L is a set of class labels. A sequence classification model $C : \mathbb{R}^{n \times m} \rightarrow L$ will then map from an input sequence X to a class label y. Email spam detection and genre classification are common use cases of sequence classification models in NLP and MIR, respectively.

EMP generation is a more complicated process. It involves mapping an input sequence (score) to another output sequence (performance). We call such a task a sequence-to-sequence (*seq2seq*) model. In this case our input data X is the same, but our output data is also defined as a sequence of vectors $Y = \{y_1, y_2, y_3, ..., y_{\hat{n}}\}$, where $y_i \in \mathbb{R}^{\hat{m}}$. We can then define a *seq2seq* model as $S : \mathbb{R}^{n \times m} \rightarrow \mathbb{R}^{\hat{n} \times \hat{m}}$ which will produce an output sequence Y given the input sequence X. In EMP generation, m is the number of score features, n is the number of input score notes, $\hat{m}$ is the number of performance features, and $\hat{n}$ the number of output performance notes.

3.2 Case Study: Neural Machine Translation

RNNs and their common adaptations as an LSTM have historically been the default modeling choices for sequential data Deep Learning footenoteFor brevities sake, we do not provide the detailed mathematical definition for RNNs here and refer to the reader to [15]. However, in recent years the Transformer[33] architecture model has outperformed RNNs in many tasks and is becoming the de-facto standard for sequential data modeling in modern Machine Learning. To provide context for the origin of the Transformer, we will outline the historical progress of an NLP task known as neural machine translation (NMT).

Machine translation (MT) is the task of computationally translating one natural language to another[1]. Traditional MT systems relied on complicated rule sets and decoding algorithms stitched together to create a statistical model known as a statistical machine translation (SMT) system. Rather than an amalgamation of several different systems developed independently, NMT systems are trained end-to-end inside a single ANN architecture and significantly reduce building MT models' complexity. NMTs are *seq2seq* models and typically operate by translating a single sentence at a time. Until the advent of the Transformer, NMT models primarily used RNNs.

One of the complexities in building an NMT model is that the source and target sentences are often not the same lengths - $n \neq l$ in our definition of *seq2seq* models given in section 3.1. NMT translation systems use what is known as an *encoder-decoder* architecture, which account for the variable-length input and output sequences. From a probabilistic perspective, the job of the *encoder-decoder* architecture is to model the conditional probability distribution of a variable-length output sentence Y given a variable-length input sequence X.

$$P(y_1, y_2, ...y_n | x_1, x_2, ..., x_l)$$

In the *encoder-decoder* architecture, calculation of the distribution is decomposed into two separate models. The encoder's job is to read in the source sequence and find a good representation, or encoding, of that sequence. In the original formulation of the *encoder-decoder* NMT architecture, Cho et al. [6] present this encoding in the form of a fixed size vector c. The decoder is an autoregressive language model[2], and uses c as a condition to

[1] Google Translate is one successful commercial application.

[2] Autoregressive models are sequence-based models that take as input the output of the model at previous time steps. Language models are instances of autoregressive models that are capable of generating novel texts of varying lengths. Language models can generate novel text from scratch, although they are often prompted with existing text to write in a particular style or on a certain subject. See this online playground for an example

generate the new sentence in the target language. Using this decomposition, we can view the decoder as calculating the probability of the next word in the sentence given all of the previous words, and the hidden encoding vector [2].

$$P(Y) = \prod_{\hat{n}=1}^{n} P(y_i|y_1, y_2, ..., y_{i-1}, c)$$

This RNN based *encoder-decoder* model improved upon the state of the art results for existing SMT based translation systems. However, there is an inherent limit imposed on the system for long sentences. Cho et al. [6] shows that the performance of a basic *encoder-decoder* model deteriorates rapidly as the length of an input sentence increases. To account for this Bahdanau et al. [2] present what is known as the *attention* mechanism. Instead of using a fixed-sized vector encoding, *attention* allows the decoder model to search for a set of positions in the source sentence where the most relevant information is concentrated and uses this information as it generates text in the target language. In simpler terms, the decoder "pays attention" to the source sentence's most relevant words to find the right translation of every word at each time step.

Rather than use a fixed-length vector at every time step, the decoder is modeled as follows

$$p(y_t|y_1, y_2, ..., y_{t-1}, X) = g(y_{i-1}, s_i, c_i)$$

where g is some non-linear potentially multi-layered function that outputs the probability of y_t and $s_{\hat{t}}$ is the hidden state of the RRN at time step t. c_i is a context vector that is generated using information about the relationship between certain words in the source sentence and the next word y_t to be generated (see Bahdanau et al. [2] for the full description behind this context vector). The concept of attention is built into a set of context vectors for each time step in the generation process. By getting rid of the restriction of using a single fixed-size

vector, this attention-based model achieved state of the art results in NMT.

Since the introduction of the attention mechanism, it has been used in tandem with RNNs and other DL models to push state of the art in various sequence-based tasks, such as Question Answering, Sentiment Analysis, and Part-of-Speech tagging [5]. There a several reasons for the increase in performance; one reason being that attention allows the models to better the dependencies of the sequence without respect to their distance from each other (this is a known limitation of the RNNs) [33], another being that attention helps with parallelization in training because of the lack of constraint on the dependency of one time step to another as is the case with RNNs. In almost all cases, the attention mechanism was used in conjunction with other modeling architectures. It wasn't until the introduction of a completely new family of NN architectures, Transformers, that the attention mechanism was applied outside of any existing network architecture.

3.3 Transformers

Richard: transformer section needs a lot of work given the change of the general document outline

The Transformer model is an attention-only neural network architecture designed for sequential data modeling. The original Transformer by Vaswani et al. [33] was built for NMT as an *encoder-decoder* model. This Transformer model significantly advanced the state of the art in NMT and has since been applied to many other sequence modeling tasks, both inside and outside NLP.

3.3.1 Attention is All You Need

Both the encoder and decoder of the Transformer architecture consist of a stack of N layers. Each layer combines the attention mechanism along with a standard pointwise fully connected feed-forward neural network (FFNN). The encoder layer uses *self-attention* - attention applied in a single sequence rather than attention between an input and output sequence. In self-attention, each element in the sequence "pays attention" to other elements in the same sequence. Regular attention is used between elements in an output sequence and elements of a different input sequence. The decoder's layers use both types of attention - here the self-attention happens for all elements of the output sequence, and the normal attention mechanism is used on the outputs from the encoder layers. This model is conceptually similar to the RNN based attention model of Bahdanau et al. [2], but uses the self-attention mechanism instead of an RNN in both the encoder and decoder. As discussed, because the attention mechanism can model longer-term memory and allows for faster training with bigger models, the Transformer achieved impressive state of the art results over attention based RNNS.

3.3.2 Transformer Adaptations: BERT and GPT

Of particular interest in the new Transformer modeling domain is its powerful adaptations of the original architecture, which have been applied to many other NLP tasks besides machine translation. One such architecture, BERT (Bidirectional Encoder Representations from Transformers) uses an "encoder only" Transformer model [7].

The original Transformer was built with machine translation in mind, but several other NLP tasks could benefit from using an attention-only architecture. Some of these tasks include standard text classification, textual entailment, sentiment analysis, and question

answering . A recent trend in NLP research is the pre-training of large models on a single task, which creates generic language data representations that are fed into models trained for a specific task [31]. BERT is such a model but uses a Transformer architecture for pre-training as opposed to an RNN. BERT uses only the encoder of the original Transformer but significantly increases the number of model parameters and is trained on a massive dataset. The pre-trained representations from BERT are then used with much simpler models trained on specific tasks, some of which have already been mentioned. This approach led to a significant improvement in the state of art for general language understanding benchmarks.

Richard: Find reference

The Generative Pre-Trained Transformer (GPT) is similar to BERT, utilizing only the decoder from the original Transformer as opposed to the encoder. Like BERT, it is trained on massive amounts of data, is significantly large than the original Transformer, and is used to pre-train a generic language representation that is fed into models for specific tasks. Because GPT is a large Transformer decoder-only language model trained on an immense data corpus, it is capable of writing novel text of surprising quality [3].

The original Transformer, BERT, and GPT have all significantly improved the state of the art for many different tasks NLP. The effectiveness of attention only Transformer models in other types of sequential data modeling is still an open research question. As we will see later in this text, there are already promising results inside of the music domain , and even in other fields such as image processing [9].

Richard: Add reference to Music Transformer

[3] Samples of text written by GPT can be found online

Chapter 4

Related Work: Music Generation Models

As in other fields, state of the art models in music generation are mostly comprised of ANN models. Some music generation tasks have already seen a Transformer application, while others (such as expressive performance generation) have not. We will provide an overview of the existing state of the art in EMP generation and transformer-based music generation models. We also give some background into the data representation and feature engineering of each model.

4.1 Existing Expressive Musical Performance Generation Models

EMP generation models fit into one of two categories, rule-based and data-based. Rule-based systems use hardcoded rules derived using pre-existing musical knowledge and empirical studies involving human cognition. Data-driven models rely on probabilistic and machine learning methods to take an existing dataset of both scores and performances and use the performance data as a guide to learn the mapping between score features and performance features.

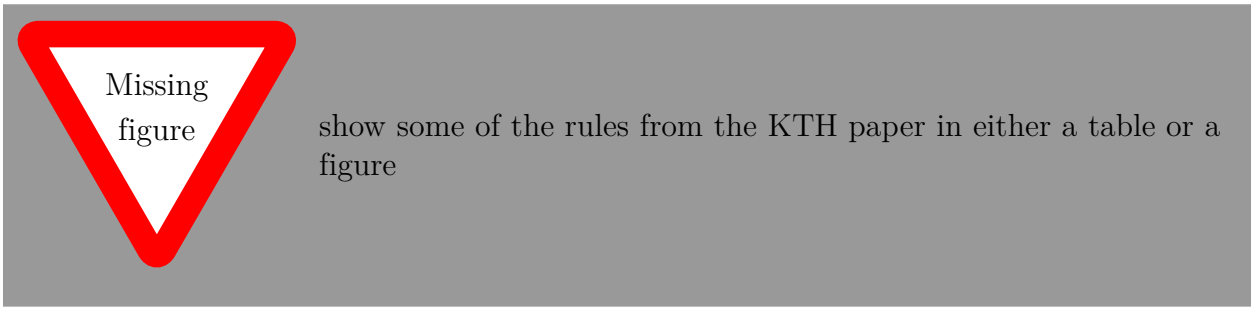

Figure 4.1: The left column shows the name of the rule, and the right column provides a language description of that rule. These are the rules that we might expect a data-based system to learn.

4.1.1 Rule Based

The KTH system [14] sits at the center of rule-based EMP models and lays the foundation for all EMP generation models. Development of the KTH started in the 1980s and has continued well into the 21st century. The KTH system's initial methodology defined rules relating to musical composition structure and how it affects a resulting performance. The first set of rules applied specifically to singing synthesis were later adapted to general musical performance.

Since then, there have been two general methods in the continued development of the KTH rule system. The first is that of *analysis-by-synthesis*, which involved using the rules to synthesize musical performances presented to human listeners (both professional and non-professional), gathering listening feedback, and then using this feedback to modify the rules where needed. The second was an *analysis-by-measurement* method. This method uses direct computation to evaluate a computational generated performance by comparing it with an existing real performance [1]. Example rules from the KTH system are found in figure 4.1 .

Richard

May need more explo-ration in caption

[1] This evaluation method is consistent with the data-driven approaches but generally applies to any generative model. Data-driven models use the performance data to directly build the model, whereas real performance data in the KTH system is for evaluation purposes only. Any further updates to the model still rely on a hardcoded set of rules

To our knowledge, the KTH rule-based system is the first sophisticated computational model for generating expressive performance. The explicitly defined rules in the KTH system may be those we expect a data-based model to learn. Widmer [34] shows that data-driven methods do learn some of the same rules as the KTH system but also learn rules that are the opposite of KTH rules. As has already been discussed, model evaluation's problematic nature may describe this phenomenon, as there is no telling which rule is more "correct" than another. Nevertheless, the KTH rule system has been an essential milestone in the evolution of EMP generation.

4.1.2 Data Based

State of the art EMP generation models rely on existing human performance data to learn the mapping between score and performance. Such models are based either on sequential probabilistic or non-linear neural network methods[4], although there has been previous work with linear and non-sequential modeling. Cancino-Chacón et al. [4] give a complete overview of all relevant EMP generation models. We will describe a few of these models and frameworks which are pertinent to our work

Basis Function Models

The first of these is a complete computational and mathematical framework for exploring EMP and is known as the Basis Model (BM) framework [10]. The BM framework for EMP describes the full end-to-end process involving both the generation and analysis of musical performance. The name "Basis Model" is derived from the definition of Basis Functions, which create score features. The BM framework also defines *expressive parameters*, which are analogous to our definition of performance features as outlined in section 2.2. Given

score features defined by a set of basis functions and a set of expressive parameters that quantify a performance, the BM framework also provides models that can map the score features to expressive parameters[2].

Eduardo [10] outlines the full mathematical definition of the BM framework and the evolution of the framework and its application with specific feature and model definitions. BM models first started as simple linear non-sequential models that learned the relationship between a set of defined basis functions and a single expressive parameter, such as MIDI velocity. This version of the BM models each expressive parameter independently from all others and implies that one expressive parameter's interpretation will not affect the other. Although this is not necessarily the case in actual performance [3], it is a simplifying mathematical assumption that makes the models' development and interpretation simpler. Both standard least squares regression and a probabilistic Bayesian approach are used to model the linear relationship.

The BM framework introduced both non-linear and sequential models, both in the form of ANNs, as its development progressed. A standard Feed-Forward Neural Network (FFNN) added the capability for non-linear modeling. This model showed an increase in both the goodness-of-fit and predictive accuracy over the linear model. A standard RNN was used to implement the sequential model with features where the time-dependent and sequential nature of music was relevant. Used in conjunction with a FFNN, this model performed the best relative to all other models.

[2]The BM framework is a mathematical outline of the components involved in EMP. The Basis Mixer is an open-source implementation of the BM framework and is made available on github

[3]For example, the effect of ***crescendo*** marking may have an affect on both the dynamics and the tempo at the same time

VirtuosoNet

VirtuosoNet, similarly to the BM framework, is a complete end-to-end system for training EMP models and generating novel performance. Although VirtuosoNet does not explicitly define a clear mathematical framework and abstract representations of EMP as does the BM framework, it nevertheless contains all of the moving parts necessary to train a EMP model from scratch. We make a distinction between VirtuosoNet as a set of NN-based EMP models and the VirtuosoNet framework, which comes with all of the necessary data processing and featuring engineering as well as model training.

Similarly to the BM framework, the development of the VirtuosoNet framework is gradual (although the general model architecture is the same). The model comprises three parts; a score encoder, a performance encoder, and a performance decoder. The score encoder learns a score representation C, which is a sequence with the same length as the input notes. The performance encoder's job is to learn the style of a particular performance. This style guides performance generation so that different performances are rendered given the same input, depending on the learned style. The performance decoder is a generative model which uses the score encoding C and the style vector z to generate the resulting performance. VirtuosoNet framework also defines both input and output features. Input features contain standard score information such as pitch, timing, key, and metric information. The VirtuosoNet framework uses these features as well as more detailed note level information such as the duration of rests, articulation markings, and others. Output features include tempo as BPM, note onset deviation, MIDI velocity, and articulation. VirtuosoNet framework, unlike any other EMP generation model, uses features to model the pedal directly. They include pedal value and note onset, pedal value at note offset, minimum pedal value between note onset and note offset, and minimum pedal value between note offset and the next note's onset. A full list of data features is given by Jeong et al. [23].

The first version of the model presented by Jeong et al. [20] is a more primitive model and forms the basis for later versions. It introduces a recurrent hierarchical attention network (HAN) made up of hierarchical LSTM layers used in conjunction with the attention mechanism. The HAN forms the building block for all of the different components of VirtuosoNet. This model's dataset is an early version of the KAIST dataset and consists of Chopin performances taken from the Piano-e-competition, with 25 compositions and 217 full performances. No quantitative or qualitative experiment results are given in the first formulation.

The next iteration of VirtuosoNet builds from the initial HAN layers but also uses a novel graph-based data representation and model. This model is defined as an iterative sequential graph-based neural network (ISGN). The graph-based NN exists in conjunction with hierarchical attention layers in both the encoder and decoder. The score is represented as a graph where the nodes are the individual notes. The graph's edges define relationships between the notes; one example is an edge indicating two notes with the same onset (which implies they belong to the same chord), another indicating notes belonging to a slur. A gated graph neural network (GGNN) is used to model the note-level graph representation and is combined iteratively with a HAN used to model measure-level features. The latest and best-performing version of VirtuosoNet[21] returns to the HAN only architecture. Similar to the first VirtuosoNet model, it uses an LSTM based HAN network for every different component of the architecture, explicitly defining the hierarchical models according to musical boundaries, including notes, beats, and measures. This model adds additional more abstract hierarchical models that exist to create better structure at the metrical level and preserve patterns across mid-level structures of the composition.

Both the ISGN[22] and HAN[21] version of VirtuosoNet are trained using the full KAIST dataset and the same evaluation methods; MSE for the quantitative evaluation and listening tests for the qualitative. The final version of HAN reports better MSE metrics than the

ISGN. The qualitative evaluation shows that both the ISGN and HAN perform better than baseline models and better than a "deadpan" performance[4]

Richard: An introduction of deadpan performances might work better in qualitative evaluation background

. The final HAN version's qualitative evaluation includes a comparison between the HAN and the publicly available version of the BM framework model [5].

The results in [21] show that the HAN performs better than the BM model. Many plausible reasons may explain the difference in results other than the HAN being a superior model to the BM. These include differences in the training data for both models, a bias of the qualitative method towards the HAN, and the fact that the listening test members' opinion doesn't necessarily imply one model being "superior" to another. However, given the results presented by Jeong et al. [21], we will assume that this version of the HAN represents the current "state of the art" in the field (or as close to it as is possible).

Richard: Add section that talks about the features used for virtuosoNet

4.2 Music Generation with Transformers

The Transformer has seen some application in music generation. Such models use a discrete representation of music as a sequence of events with highly sparse encoding vectors. This representation is consistent with the word embeddings used to train most NLP models and so has a natural extension in Transformers, designed with language data in mind.

[4] A deadpan performance is one generated using the rule-based systems in most musical annotation software. The "deadpan" performance is one that sounds robotic, mundane, and without much musical expression

[5] The website for the BM model can be found https://basismixer.cp.jku.at/static/app.html. At the time of this writing, the website is currently unavailable

The first model to use the Transformer for music generation is known as the "Music Transformer" [19]. It directly builds from the PerformanceRNN of Oore et al. [29]. The PerformanceRNN uses the Piano-e-compeitition data to directly generate MIDI. This model *simultaenously generates a composition and a performance*. The MIDI format encodes both aspects of a score such as the timing and pitch of notes but also elements of expressive performance such as dynamics and pedaling (see appendix). PerformanceRNN uses an event based representation of MIDI and all input tokens are a one hot encoding over the different possible MIDI events. There are 128 possible NOTE-ON events, 128 possible NOTE-OFF events, 125 possible TIME-SHIFT events, and 32 possible VELOCITY events, which leads to a 413 dimesional one hot encoded vector. PerformanceRNN is an autoregressive LSTM model which models the probably of generating the next note given all of the previous notes, similar to a language model.

Richard add ref

Music Transformer uses the original Transformer architecture and implementation of Vaswani et al. [33] with some adaptation, and applies it to the same data set and representation of the PerformanceRNN. One of the important parts of the original Transformer architecture is a positional embedding. This embedding encodes the sequential nature of the data into the model which is otherwise not accounted for using only the attention mechanism. The original positional embedding is referred to as an "absolute" position, meaning that the embedding only encodes information about the position of each element in relationship to the beginning of the sequence. Music Transformer uses a "relative" position, which instead adds information about how far apart two positions in the sequence are. The relative positional embedding accounts for every possible pairwise distance of each element and this information is then processed by the attention mechanisms throughout the rest of the model. It allows attention to account for the distance of any two notes to each other, rather than the global position of a single note in the sequence.

The Transformer architecture with the relative positional embedding outperforms other baseline models, including the LSTM based PerformanceRNN. Qualitative evaluation shows that the Music Transformer generates performance with much better long term structure and overall musical cohesiveness than the performances from the PerformanceRNN[6]. This result is consistent with the observation that attention has a better memory than the hidden state of an RNN cell.

Building from the Music Transformer, Open AI introduced MuseNet [30]. MuseNet is similar to GPT both in terms of the model architecture and philoshopy. It uses a very large decoder only Transformer model and trains it on a massive amount of data to build a model that is capable of generating novel music across many different domains. MuseNet is trained using only MIDI data gathered from various sources of the internet (including the MAESTRO dataset) and uses an event based token represenation similar to the Music Transformer. In contrast to the Music Transformer which only operates wester classical solo piano music, MuseNet is capable of generating music across a variety of genres in multiple instruments. Open AI has not directly released any research results comparing the benchmark results of MuseNet to other models, but it is evident from listening to MuseNet samples[7] that it generates high quality music.

Open AI also released music generation model which uses Transformers, JukeBox [8], that deals directly in the audio domain. The choice to model directly in audio is motivated by the fact that symbolic forms such as MIDI and MusicXML do not capture the subleties of musical performance that are crucial in the actual experience of a musical listener. JukeBox is comprised of a Vector Quantized Variational AutoEncoder (VQ-VAE) whic learns to encode the audio data into a lower dimensional representation, and a Transformer model similar to

[6]Sample performances for both the PerformanceRNN and Music Transformer are available online from the Magenta Research group of Google AI.

[7]https://openai.com/blog/musenet/

MuseNet which uses the encoding to generate new music. Although the samples generated by JukeBox are impressive both for their sound quality and overall musical cohesiveness, much work is still needed in the direct audio generation domain to produce audio of high enough quality that it competes with human productions.

Chapter 5

Methods and Experiments Results

Given the relevant background research and knowledge base, we will now describe the experiments we ran and the reason behind our experimental methods. Given the powerful advances in NLP and music generation tasks, our general goal was to investigate the Transforme model in application to EMP generation. Both language and music are by nature sequential and hierarchical - as such, one would expect the strengths of a Transformer model created with NLP in mind to have natural extensions into a diversity of musical tasks. We use the general framework for a complete end-to-end performance generation system of VirtuosoNet. This project's initial purpose was to determine if a Transformer-based model improves upon VirtuosoNet, given the same data, features, and evaluation metrics. However, as discussed in section 6, answering that question proved to be more complicated than initially anticipated and led our research in a different direction. We present our initial method and experiment results in this chapter.

5.1 Data and Features

The reasons for adopting the VirtuosoNet system are twofold: the first being that the KAIST dataset is the largest publicly available dataset used in EMP generation, the second that the data processing and model architecture code VirtuosoNet are open-sourced[1]. If we consider

[1]https://github.com/jdasam/virtuosoNet

VirtuosoNet to be the state of the art in EMP generation, it provides a natural starting place to compare against any further model development. The VirtuosoNet system uses handcrafted features for both scores and performances, which are outlined in section 4. We use the same feature set.

5.2 Model

In the VirtuosoNet system, there is a 1:1 mapping between notes in scores and performances. As mentioned in 3, the original Transformer is an encoder-decoder *seq2seq* model with variable-length input and output sequences and accounts for this complexity by creating a generative decoder model which relies on the input encoding. Because our input and output sequences are the same lengths, our model is a simpler encoder-only Transformer model, conceptually similar to BERT. Sitting on top of the Transformer encoder is a simple FFNN that uses the learned representation from the Transformer and maps to the final output feature set. We use the absolute positional embedding of Vaswani et al. [33] used with the original Transformer.

5.3 Experiments and Model Evaluation

Our initial experiments were designed to answer two questions:

1. Does a Transformer based EMP generation model outperform a similar LSTM based EMP generation model?

2. Does a Transformer based EMP generation model outperform existing state of the art (VirtuosoNet) models?

To answer both of these questions, we ran all of our experiments using the existing VirtuosoNet framework. The VirtuosoNet framework defines real-valued features and the resulting models are regression models. All VirtuosoNet models use MSE as both the loss function for training and reconstruction error for quantitative evaluation. VirtuosoNet models are trained using an 8-1-1 train/valid/test data split, and Jeong et al. [21] present MSE results for each different parameter of the performance features on the test set. We follow the same method for our quantitative evaluation. Our models were trained at 50 epochs, and the best model parameters were selected according to the lowest validation evaluation score. Our models were implemented in PyTorch using their native Transformer library. We used Neptune AI [27] to manage our experiments and report the metric feedback. Data for all of our experiments, including the model hyperparameters and metrics can be found online[2].

Our main experimental goal was to find the best model configuration for a Transformer based model. To do so we ran a fairly exhaustive hyperparemeter search for our Transformer model. To guide our search we trained in tandem with our Transformer model our own baseline LSTM model, as well as other VirtuosoNet models. We used the evaluation results from the validation set to gain an intuition about which family of models performed the best. The final test set evaluation happened after all model development was finished.

5.3.1 Quantitative Evaluation

We use several different model configurations for the Transformer. Our Transformer baseline has 6 layers, 6 attention heads, and a hidden size of 256. We chose this as a base configuration because it closely matches the size of the original Transformer [33], except for the hidden size of the feed-forward layer. We chose a smaller hidden size of 256 for our base layer hidden size due the relatively small dimensionality, 78, of the input data to the model. We trained from

[2] http://ui.neptune.ai/richt3211/thesis

scratch the Iterative Sequential Graph Network (ISGN) and the HAN baseline (HAN-BL) models as reported in [22] and [21] respectively and used the metrics from the validation data set to guide our own model development before we ran the final evaluation. Table 5.1 contains the result of the final evaluation.

The first major implication of the results is that the Transformer model easily outperforms the baseline LSTM, but does not outperform the VirtuosoNet models. Within the Transformer models, the best performing model uses all of the same baseline Transformer parameters with a major increase in the dimensionality of the hidden layer (model $T_{N_{133}}$). Other models that perform well include those which only increase the number of attention heads ($T_{N_{132}}$ and $T_{N_{26}}$) and those with an increase across several different paramters ($T_{N_{135}}$). It appears that increasing the number of layers degrades performance ($T_{N_{118}}$, $T_{N_{181}}$, $T_{N_{125}}$), and so does increasing the droupout and learning rates ($T_{N_{171}}$, $T_{N_{173}}$).

Although we do see improvements by increasing the hidden dimension and number of attention heads, the performance improvements aren't significant. There is only a difference of 0.03 in MSE between the best performing Transformer model and the Transformer baseline, whereas there is a difference of 0.11 between the best Transformer model and the best HAN model. From our initial interpretation it would seem that a Transformer model can't even match the existing state of the art in EMP generation (let alone beat it), but as we discuss in later sections, this interpretation of results is not necessarily valid.

Model Configuration								Results in MSE					
N_{id}	M	L	d_{hid}	D	LR	C	H	Tot	t	v	d	a	p
123	LSTM	3	256	0.1	0.1	0.5		1.08	0.84	1.35	1.02	1.11	1.08
147	T-BL	6	256	0.1	3e-5	0.5	6	0.86	0.54	0.80	0.88	0.80	0.92
169			128					0.87	0.55	0.79	0.96	0.88	0.92
128			528					0.86	0.50	0.76	0.88	0.82	0.93
133			1024					0.83	0.47	0.76	0.88	0.82	0.88
118		12						0.89	0.65	0.82	0.88	0.82	0.95
181		24						0.93	0.62	0.97	0.89	1.09	0.95
132							13	0.84	0.51	0.77	0.95	0.81	0.88
171				0.2				0.91	0.74	0.82	0.95	0.86	0.94
173					0.01			1.01	0.86	1.05	0.90	1.18	1.01
188							26	0.84	0.62	0.78	0.89	0.79	0.87
134		12	528					0.85	0.54	0.75	0.90	0.86	0.89
190		12					13	0.87	0.49	0.78	0.89	0.87	0.94
135		12	528				13	0.84	0.47	0.81	0.89	0.86	0.89
125		24	528					0.93	0.69	0.99	0.91	1.12	0.94
	HAN-BL	-	-	-	-	-	-	0.77	0.40	0.67	0.77	0.72	0.84
	HAN-S	-	-	-	-	-	-	0.73	0.27	0.61	0.75	0.69	0.82
	HAN-M	-	-	-	-	-	-	0.72	0.22	0.53	0.75	0.75	0.81

Table 5.1: A comparison of 3 different families of EMP generation models: virtuosoNet models, Transformer models, and our LSTM baseline models. The left side of the table presents the configuration for each of the models, exluding the virtuosoNet models which are present in other works [21, 22]. N_{id} is the ID of the Neptune experiment, L is the number of layers, d_{hid} is the dimension of the hidden layers, D is the dropout, LR is the learning rate, C is the gradient clip, and H is the number of attention heads. The right side of the table presents the MSE results for all models along the five different expressive dimensions mentioned in 6.1, as well as the total MSE which is an aggregation of all the individual expressive features. The entries for the HAN models come from virtuosoNet and are given in [21]

Chapter 6

Analysis

Our initial experiments using a quantitative evaluation indicated that our Transformer model does not improve the state of the art in EMP generation. As a method of ”debugging” our Transformer model to look for potential errors and improvements, we wanted to actually listen the performances generated by our models to identify any problems, if they existed at all.

These listening tests acted both as an error analysis as well as our qualitative evaluation of the models. It became apparent to us right away that there was a disconnect between the quality of the model according to the quantitative metric and our own evaluation while listening to performances. In this chapter, we present this analysis. It includes the method used for our qualitative evalution, new experimental methods to improve our own models, and conclusions about the quality of each model according to our listening tests.

6.1 Identifying Training Problems

To conduct our listening tests, we first had to use the trained models to generate performance MIDI files. We generated performances for a variety of our trained models using six compositions from three different composers, listed in table 6.1. The MIDI performance files are available for download using the Neptune AI web interface. For easy comparison of each model's performance, we loaded their relevant MIDI files into the Digital Audio Worksta-

Composer	Composition
Bach	Prelude in E Minor, BWV 855
Bach	Prelude in F-sharp Major, BWV 858
Chopin	Etude Op. 10, No. 12
Chopin	Fantaisie-Imropmptu
Beethoven	Piano Sonata No. 17 First Movement
Mozart	Piano Sonata No. 11 First Movement

Table 6.1: The compositions used for the qualitative evaluation of our models. All scores come in the form of MusicXML from MuseScore. None of the scores were present in the training data

tion (DAW) Logic Pro X. We used their "Steinway Grand" instrument synthesizer to create the final audio. Logic Pro X also provides visualizations of different performance aspects, including the MIDI velocity of each note and the sustain and soft pedals' value throughout the performance. [1]

Richard: Add reference to image with logic pro x visualizations

. We used these visualizations in tandem with our interpretation of the generated audio to guide our analysis.

As mentioned earlier, it was apparent that there was a mismatch between a model's performance quality according to the MSE value and our evaluation. For example, the Transformer model with N_{id} 125 (which we will denote as $T_{N_{125}}$, see Table 5.1) had much worse validation MSE metrics than almost every other model Transformer model. However, our listening test revealed that the absolute tempo for smaller models, such as the Transformer baseline $T_{N_{147}}$, was much faster and sounded worse (to the point where the performances are almost 'unlistenable') than $T_{N_{125}}$. The potential disconnect between the 'quality' of the model as

[1] Performances generated by models can be view through our Neptune Project. Each experiment has an ID and we've run performance generation code for many of the models. To listen to performances, visit an experiments' artifacts tab and download available performance MIDI files. These MIDI files can be played in any DAW software, such as Logic Pro X. If performances don't exist for an experiment, contact the autor

determined by quantitative and qualitative evaluation led us to investigate possible problems with the training methods used by Jeong et al. [21].

One of the first potential problems we identified was the method used to calculate and interpret the loss and evaluation. The output features of virtuosoNet are a sequence of vectors with a length of 11. The first four features correspond to a single component of expression. They are: tempo, velocity, deviation, and articulation, respectively. The last seven features are all different numbers that correspond to information about the pedal [23]. Jeong et al. [21] present MSE metrics for five different expressive parameters, which include all of those previously mentioned, as well as the pedal. Both Jeong et al. [21] and our references to pedal MSE are an aggregation of the seven different features that contain pedal information. In contrast, the other 4 MSE metrics only apply to a single feature.

The original MSE used to train virtuosoNet assumed that every feature of the output vector contributed equally to the final output and corresponding loss optimization. Given that there is seven times more information for the pedal parameter than all others, we can conceptualize what we will call VirtuosoNet framework loss as placing more importance on the pedal than every other expressive feature. Given that the VirtuosoNet framework MSE loss is used both for model training and evaluation, this means that it inherently rewards those models which express pedal better than all other performance features. If the VirtuosoNet models outperformed our Transformer models according to this metric, does that mean that they are better at modeling EMP generation as a whole, or that they are simply more fit to model the pedal? To answer this question, we came up with a new weighted MSE loss function that allows for a more fair representation of the evaluation method.

We define the output vector as an 11 dimensional vector $\mathbf{v} = \{t, v, d, a, p_0, p_1, p_2, p_3, p_4, p_5, p_6\}$ where t, v, d, and a represent tempo, velocity, deviation, and articulation respectively, and p_i represents a single component of the pedal. For a predicted output vector $\mathbf{v}$ and the target

output vector $\widehat{\mathbf{v}}$, standard MSE loss is $MSE(\mathbf{v}, \widehat{\mathbf{v}}) = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{v}_i - \widehat{\mathbf{v}}_i)^2$. This can also be re-written as $MSE(\mathbf{v}, \widehat{\mathbf{v}}) = \frac{1}{11}[(\mathbf{v}_t - \widehat{\mathbf{v}}_t)^2 + (\mathbf{v}_v - \widehat{\mathbf{v}}_v)^2 + (\mathbf{v}_d - \widehat{\mathbf{v}}_d)^2 + (\mathbf{v}_a - \widehat{\mathbf{v}}_a)^2 + \sum_{i=1}^{7}(\mathbf{v}_{p_i} - \widehat{\mathbf{v}}_{p_i})^2]$. We introduce 5 different weight values: $\alpha_t, \alpha_v, \alpha_d, \alpha_a$ and α_p. Our weighted MSE loss is defined as $W_{MSE}(\mathbf{v}, \widehat{\mathbf{v}}) = \frac{1}{\alpha_t+\alpha_v+\alpha_d+\alpha_a+\alpha_p}[\alpha_t(\mathbf{v}_t - \widehat{\mathbf{v}}_t)^2 + \alpha_v(\mathbf{v}_v - \widehat{\mathbf{v}}_v)^2 + \alpha_d(\mathbf{v}_d - \widehat{\mathbf{v}}_d)^2 + \alpha_a(\mathbf{v}_a - \widehat{\mathbf{v}}_a)^2 + \alpha_p\sum_{i=1}^{7}(\mathbf{v}_{p_i} - \widehat{\mathbf{v}}_{p_i})^2]$. The original MSE can be seen as the weighted MSE with $\alpha_t, \alpha_v, \alpha_d, \alpha_a = 1$, and $\alpha_p = 7$.

According to our formulation of the loss, the pedal's optimization is weighted much higher than every other parameter for the original VirtuosoNet framework loss. Is this is the right way to conceptualize what a 'good' performance is? Would a different configuration of the expressive feature weights lead to a better outcome? Answering these questions is non-trivial. Therefore, we call to question the validity of using MSE as an evaluation metric, especially as presented in the VirtuosoNet framework. With this in mind, we ran additional experiments changing the weights for each expressive parameter to see what the effects on performance would be according to our evaluation.

6.2 Qualitative Evaluation

Our first general observation given our new loss evaluation is that the two most important factors for overall performance are tempo and pedal. Models that don't perform either of these two features within certain bounds correctly make performances unlistenable. Suppose a performance's global tempo is too fast, and every other expressive parameter renders correctly. In that case, the resulting performance will still sound bad enough that it's not worth listening to at all [2]. We noticed a similar phenomenon with the pedal. Some models generated performances with the sustain pedal applied at all times with very few breaks. The

[2] See $T_{N_{86}}$ Fantaisie Impromptu and $T_{N_{126}}$ Etude Op. 10 No. 12.

result is a performance that sounds "muddied" and unrefined. Although these performances are more bearable than those with extreme tempo, they are still hard to listen to in any meaningful way [3].

We also notice that the tempo and timing of the Transformer models are more dynamic than the LSTM models, both for our LSTM baseline and the VirtuosoNet models. For some models, the variability in timing seemed to be a good thing, while for others, it was so bad that it almost sounded like the model was still "learning" how to play. The tempo for all LSTM based models (except for some slight variations in the performances from HAN-M [4]) was extraordinarily consistent and non-changing to the point of sounding robotic and mundane. On one extreme with Transformer models, the highly dynamic tempo at times sounds like a real performer making mistakes [5]. The other extreme is performances of some LSTM models that are so boring they don't sound "human" at all [6].

The pedaling in general of all models was mediocre at best. This observation is consistent with that of Jeong et al. [21]. There are models whose performance pedaling is much better than others and follows the music's natural cadence but still don't match pedal in actual human performance. Figure 6.1 shows a visual comparison of the sustain pedal usage in different performances.

The importance of tempo and pedal was part of the intuition that led to our formulation of the weighted MSE by expressive parameter defined in 6.1. Our first experiments with the weighted MSE used an even weight distribution. Because changing the loss function also changed our evaluation, we could not directly compare the models' quantitative results, and so no quantitative values are reported. All evaluation was by our qualitative listening test.

[3] See $T_{N_{125}}$ Piano Sonata 11
[4] Performances for the HAN-M are available at N_{126}
[5] See $T_{N_{86}}$ Piano Sonata 11
[6] See $LSTM_{N_{123}}$ Piano Sonata 11

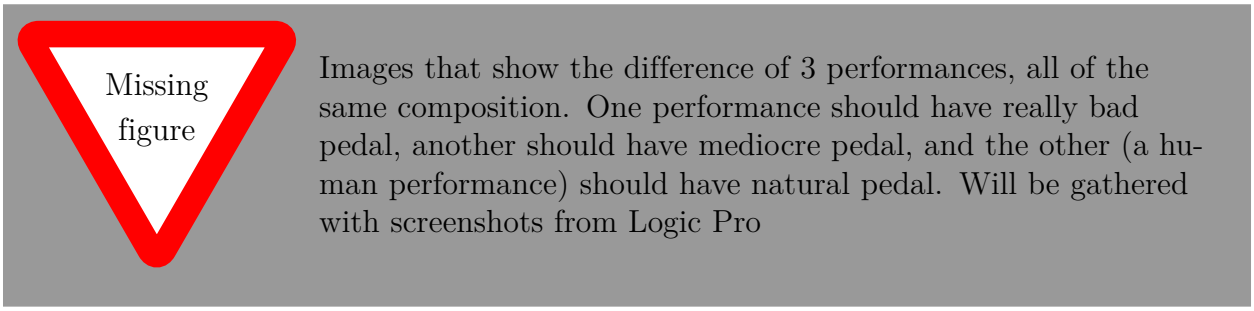

Figure 6.1: Test Caption

We ran a few additional experiments with the tempo and pedal weighted high and some other changes in the model size. Table 6.2 shows a full description of the models and their parameters.

$T_{N_{154}}$, which weights all expressive parameters, generates performances with the global tempo slightly too fast and an extremely muddy pedal. $T_{N_{150}}$ which uses the original MSE loss with a higher pedal weight, produces better performances with reasonable pedaling. However, the tempo is inconsistent enough that the performance loses its cohesiveness as a whole. For these reasons, we increased both the tempo and pedal weights to be much higher than the others in models $T_{N_{156}}$ and $T_{N_{157}}$. We found that the tempo and pedal weights $T_{N_{156}}$ were a bit too low - specifically, the pedal is almost just as muddy as it is in $T_{N_{150}}$ and the tempo is still a bit too fast, albeit more consistent and cohesive. We found that $T_{N_{157}}$ produced the best overall performance in the Transformer based models[7]. It is likely that were we to continue to experiment with a different configuration of the weights that we could come up with even better results.

Our last general observation is that the VirtuosoNet model HAN-M produces the best overall performances. In general, we feel that the tempo for HAN-M is a little too slow, but it still creates the most natural expression. This expression is most apparent in its performance of

[7]Performances of Fantaisie Impromptu best demonstrate these differences. Compare $T_{N_{154}}$, $T_{N_{150}}$, $T_{N_{156}}$, and $T_{N_{157}}$

Model Configuration					Expressive Weights				
N_{id}	L	d_{hid}	H	AM	α_t	α_v	α_d	α_a	α_p
150	256	6	6	a	1	1	1	1	7
				p	1	1	1	1	7
154					0.2	0.2	0.2	0.2	0.2
156					0.33	0.11	0.11	0.11	0.33
157					0.4	0.067	0.067	0.067	0.4
				p	0.4	0.067	0.067	0.067	0.4
159	528	12	13		0.4	0.067	0.067	0.067	0.4

Table 6.2: The model configurations of additional experiments we ran after our initial quantitative evaluation effort. We show similar hyperparemters as in table 5.1, with an additional parameter AMwhich reprsents the articulation mask. A value of 'a' indicates that the articulation value was masked according to the note alignment, and a value of 'p' indicates that the articulation value was masked according to the pedal status. There are additional paramter values that are not present but are used in table 5.1: LR is 0.0003, C is 0.5, and D is 0.1

Beethoven's Piano Sonata 17, whose introduction leaves a large space for interpretation to achieve the desired listening result - rendering the score precisely as it is written results in an uninteresting performance. We found that the only model that made this performance "interesting" was the HAN-M. Although we have previously emphasized the problem using the existing quantitative metric to evaluate our models, our quantitative and qualitative evaluation of the HAN-M model indicates that it is the "best" model. The proposed Transformer architecture does not improve upon existing models. We will provide some intuition about why this is and possible model improvements for future work in section .

Richard

Add ref to discussion

Chapter 7

Discussion

Chapter 8

Conclusion

As we have outlined, this project underwent an unexpected evolution of purpose. Our initial goal was to determine if we could outperform the existing state of the art system in EMP generation with the exact same research methods and only a change of computational model. The initial quantitative experiment results indicated that a Transformer does not outperform hierarchical based recurrent models. However, when we ran our own subjective evaluation of the comparison in performance among the two families of models, we determined that the definitive declaration of one model being better than another was not so straightforward. This led us to question the validity of the original research method, particularly with the evaluation metric, and shift our own toward more nuanced and discovery-based experiments. The experiments themselves were not guided by any strict method, and as such, we cannot provide results that we consider to be robust or reliable. However, we would like to use our experience in running these experiments to provide some suggestions for the direction of future work in the area.

8.1 Directions for Future Work

Our two general suggestions for research are centered on the main contributions of this work; that is, modeling and evalution. Although our Transformer based model was "outperformed" by the existing recurrent model, we feel that the Transformer model family still has the

potential to improve upon recurrent models given the right architecture. As far as evaluation is concerned, we use our experience in our model development to consider the possbilities for what a better evaluation metric might look like.

8.1.1 Modeling: Performance as a Generative Process

In section 5.3 we present our proposed Transformer model and the reasoning behind the model selection. This reasoning was based upon the fact that the job of our model was to learn the one-to-one mapping between a single note in a score and it's corresponding expression in a performance, as well the success of similar Transformer adaptations such as BERT[7]. This is in contract to the original encoder-decoder Transformer architecture, whose purpose in machine translation is to learning the mapping between *variable* length sequences. In the encoder-decoder architecture, the job of the encoder is find some good representation of the input data (the original language) that it presents the decoder, who uses this data representation along with it's own internal representation of the output data (the translation language) to *generate* from scratch the target sequence. Although for our particular formulation of EMP generation (based on the existing feature design of virtuosoNet) there is a one-to-one mapping between a score note and a performance note, we believe that the encoder-decoder architecture would be more appropriate for performance generation than our proposed encoder only model. The former assumes that *generation on the part of the performer* is as fundamental a component of the performance process as is understanding the score. The latter assumes that performance itself is only a matter of correctly understanding and rendering the information presented in the score, and doesn't learn the fundamental aspects of performance itself. Given that our model implements the latter, we believe that explains both the lack of expression and perceived error in creating novel performances, as well as the lower performance according to the quantitative metric.

virtuosoNet itself is based on an encoder-decoder architecture, along with the option to encode a specific performance style (which we feel is useful, but not necessary for performance in general). It uses a combination of pre-defined musically informed hierarchical boundaries (starting from measure, to beat, and ending in a single note) along with the attention mechanism to build both an LSTM based score encoder and generative performance decoder. We believe that it is the encoder-decoder architecture that explains the different in performance between virtuosoNet and our proposed model, and not necessarily indicative that recurrence mechanisms (LSTM) outperform attention mechanisms (Transformer). We suspect that a full encoder-decoder Transformer architecture would have the capability using attention alone to learn the hierarchical boundaries of music that are handcrafted into virtuosoNet. Because the Transformer is more unsupervised from that perspective, it's possible that it can learn additional hierarchical levels that are important in performance. Such a model (whose results may or may not outperform a recurrence based model) would be more useful from an analytical perspective, not only a generative one.

It would also be useful to experiment with the positional embeddings and relative attention mechanisms that are part of the Music Transformer [19] which is used to generate both composition and performance together (see). The increase in direct performance generation of the Music Transformer using relative position attention as opposed to absolute position based attention would likely apply to EMP generation based on a score.

Richard

Add section about Music Transformer

The implementation of a full Transformer and the experimentation with relative attention are the next immediate steps for future work. We did attempt to implement the full Transformer model but ran into practical training problems using the native implementation of the Transformer in PyTorch, which is heavily based upon classification of text data which uses highly dimensionional word embeddings. We plan to continue work in this area using custom adaptations and possible innovations of the original Transformer, using the knowledge

gained from this work to guide our experiment development.

8.1.2 Evaluation: Towards Better Metrics

As has been lengthily discussed, we believe current quantitative methods for EMP generation are in need of significant improvement. Current methods are based upon comparing a predicted performance against an actual performance and calculating some overall numerical distance between the output features, usually as the Mean-Squared-Error. As we have brought to light, this score is going to be highly dependent on the featurization of performance expression. In our case, the performance features carried more information about the various aspects of the sustain and soft pedals than all other expressive features. Using MSE with this feature set will bias the evaluation towards models with pedal over other import features relating to tempo, timing and articulation. A different set of output features fundamentally changes the interpretability of the model and makes the comparison of models with different output features impractical.

A direct comparison of a predicted performance with a single target performance will also create a strong bias toward the human performers interpretation in that performance. An evaluatoin on a large scale such dataset such as ours that has multiple performances for a single score will naturally account for some of this bias by presenting multiple "correct" interpretations for a single score and rewarding those models which can create performances that have commonalities between them all. However, the extent to which this bias exists is difficult to account for in interpreting evaluation results. This is especially dependent on the performances that exist in the evaluation set. For example, two of the scores in our test data set are Bach's Prelude and Fugue in F Major (BWV 858) and Chopin's Etude Op 10. No 2. There are 2 performances of Bach's Prelude and Fugue and 11 performances of

Chopin's Etude. Does this mean that evaluation will require more generality for the models performance of Chopin's Etude than it will for Bach's Prelude and Fugue? If generality across composers and performance styles is desired (which for us is the case), how much can trust the model evaluation given this knowledge of the test data?

Qualitative evaluation of models is used to address the potential problems of using numerical methods to measure performance. Although the qualitative evaluation methods are also subject to their own heavy bias and potential lack of consistency across multiple experiments, they do provide the "human" element of evaluation which produces an additional level of confidence in the result of the models. Of course, qualitative evaluation methods present a slew of their own practical concerns in evalution. Not only does the diversity of musical experience and knowledge of the listeners create a large space for interpretability of the results[1], it is also difficult to gather together a group (no matter their background) of people who are willing to particpate in the listening evaluation. In our case we didn't have the time or the resources to put together such an evaluation. This in combination with the lack of confidence we can place in the quantitative metric made conducting research difficult and frustrating.

These issues point to the strong need for more standardization in the feature engineering, data sets, and evaluation methods of EMP generation models. We again draw from our comparison of EMP generation to machine translation to draw some insight on how to develop more standardized methods.

Richard: Continue discussion of better evaluation methods. Need to do more research on current methods

[1]Some evaluations present performance to experienced and profesional musicians [29], while others use student musicians [21]. A case can be made using lay people with no formal musical education as the listeners is also a valid method, considering the lack of musicologolical bias they would hold.

8.2 Looking Forward: Finding the *Essence* of Performance

As we went about conceptualizing and developing this project, one thought that has plagued us is that we don't have a proper understanding of the fundamental components that comprise musical performance. Widmer in his *Con Espressione*[2] Manifesto[35] emphasizes the importance of focusing on the finding the *essence* of music itself and using that deeper understanding to more impactful technology related to MIR. When we first started the project we thought that it would be relatively simple to throw the powerful attention mechanism at a fundamentally sequential based modelling problem and that we would see improved results. It is only after running our experiments and taking a step back to look at the results that we understand what Widmer means when he refers to the *essence* of music.

Our experience in the development of this project further convinces of the fact that the computational study of music is an inherently difficult problem, not because of the limits of computation, but because of our current limited understanding of what music actually is. It is relatively simple to experience music on a personal level and to share in that experience with others. However, it is our (and Widmer's) conjecture that there is a fundamental disconnect between our understanding of the phenomenological aspect of music and the actual statistical patterns of nature that make it so appealing. MIR research attempts to encode these statistical patterns in computation and as such, deriving results that are meaningful in practical application with real human interaction is non-trivial. Widmer's suggestion is that we focus on gaining better understanding of the relationship between music in nature and music as it is perceived, and we echo that sentiment here.

This may be an essential component of deriving better evaluation systems for performance,

[2] *Con Espression* is the Italian phrase for "with feeling" and is used as direction in musical notation

which can hopefully drive the future development of performance generation models. To create a proper evaluation system or metric, we first need to understand *what exactly it is that constitutes a "good" or "bad" musical performance to a human listener.* This is a separate question from determining the quality of a musical composition, or even of the synthesis of a performance (whether it is the form of an acoustic instrument or a digitial synthesizer). To us, it is not clear where to draw the exact line between composition, performance, and synthesis from the perspective of the human listener. Although we conceptualize them as independent from each to mathematically define our problem space separate as shown in Figure 2.1, they may in fact be entirely dependent or even the same phenomena expressed differently through nature. For example, a musician performing a Jazz improvisation on a guitar may use the physical process of synthesizing sound, such as the way he strikes the guitar string or bends the string to reach a particular note, as driving factors in the musical piece. It is the actual physical limitation of the guitar instrument as the driver of creating sound that enables him to create musical subtleties in both the spontaneous composition and performance of improvisation. Is there a clear cut line between what constitues composition, performance, and synthesis in such a case?

On the flip side, we can analyze over 1000 symbolic musical compositions by Johann Sebastian Bach and their many musical adaptations over the last several hundred years. One example is the well known adaptation of his Prelude No. 1 in C Major, BWV 846 published in 1722 as the accompaniment to a melody composed by Charles Gounod and set to the lyrics of the well known Latin prayer, *Ave Maria.* The original arragement, published in 1853 was for violin (or cello) with the piano[3], but it has since been arranged and performed countless times for different instrument including guitar, string quartet, piano solo, solo vocal and full choir. It is clear that we can view the original piano Prelude as it's own composition and

[3]For an example performance, look here for a recent performance by the well known cellist, Yo-Yo Ma

separate from the many different adaptations in composition, performance, and synthesis that complete the full musical experience.

All of this is to say a good definition for what makes the *essence* of musical performance may be impossible to define without further musical exploration. We would like to adopt Widmer's philosophy to guide our future explorations and draw from the fields of musicology, music pyschology, and music cognition as well as further advancement in computer science and mathematics to come closer to discovering this essence. From this work, we have learned that using the attention mechanism and Transformer models in EMP generation creates much more dynamic performances, both "good" and "bad", than recurrence models. The *essence* of this finding could be that using the long term memory that attention provides allows more creative freedom in the performance process than the shorter term memory of an LSTM which might be more constrained by global score features such as the overall tempo. It also may be that having too much creative freedom without respect for global conditioning breaks the inherent musical boundaries defined by congitive perception, as we found to be the case with our Transformer models which were so fast that they were "unlistenable". Further experiments with modeling, data gathering, feature extraction, and evaluation, as well as an exploration of what determines a quality musical experience from a musicological and human cognitive perspective, will help answer these questions. It is our hope that we can continue to explore these problems (difficult as they may be) using additional perspectives drawing from musical research, to come closer to finding the *essence* of performance, and music as a whole.

Appendices

Appendix A

Appendices I

A.1 Musical Concepts and Terminology

A.1.1 Pitch

The first and most basic component in music is pitch. Pitch is a perceptual property of sounds that relates to the physical frequency of a sound vibration [25]. It is what determines whether or not a sound can be though of as ”high” or low”. The most commonly known way to conceptualize pitch is the 88 different keys on a piano keyboard, where each key represents a different patch value. Pitch is most commonly labeled using scientific pitch notation, which couples a range of letters (A to G) with a range of numbers (zero to eight) that correspond to different octave ranges [1]. The most well known pitch is C4, or ”middle C”, and lays in the very center of a standard 88 key piano.

Richard

Create or find visual-ization

A.1.2 Tempo and Timing

Tempo in music describes the rate at which notes are played, and timing describes when a particular note should be played relative to the start of the composition. They are best explained in the context of modern western musical notation introduces the idea of note

[1] https://en.wikipedia.org/wiki/Scientific_pitch_notation

durations, time signatures, measures, and beats [2].

Richard: Find a more intuitive way to explain this. The piano roll explanation and visualization may work better

Each composition is broken down into a sequence of measures, and the time signature defines how many beat exist per measure, as well as the duration of a single beat. For example, a 4/4 time signature indicates that there are 4 beats per measure (the top half of the time signature), and that the duration of each beat is represented by a quarter note. A 3/4 time signature would indicate only 3 beats per measure, with the beat duration represented by a quarter note. The timing of a note would refer to it's measure, beat, and note duration. Tempo is most commonly given in beats per minute (BPM). A composition with a 4/4 signatue and a 120 BPM would mean that after one minute, 30 measures of the composition should have been played so far.

Richard create or find visualization

A.1.3 Dynamics

Dynamics can simply be thought of as how loud or soft a note should be played (or has been played).

A.2 Data Representation

A.2.1 MusicXML

A.2.2 MIDI

[2] See https://en.wikipedia.org/wiki/Musical_notation#Modern_staff_notation for a more detailed explanation

Bibliography

[1] 2002 international piano-e-competition. URL https://www.yamahaden.com/midi-files/item/2002-international-piano-e-competition.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[3] Jean-Pierre Briot, Gaetan Hadjeres, and Francois-David Pachet. Deep learning techniques for music generation–a survey. *arXiv preprint arXiv:1709.01620*, 2017.

[4] Carlos E Cancino-Chacón, Maarten Grachten, Werner Goebl, and Gerhard Widmer. Computational models of expressive music performance: A comprehensive and critical review. *Frontiers in Digital Humanities*, 5:25, 2018.

[5] Sneha Chaudhari, Gungor Polatkan, Rohan Ramanath, and Varun Mithal. An attentive survey of attention models. *arXiv preprint arXiv:1904.02874*, 2019.

[6] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[8] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford,

and Ilya Sutskever. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*, 2020.

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[10] Carlos Eduardo. *Computational modeling of expressive music performance with linear and non-linear basis function models*. PhD thesis, JOHANNES KEPLER UNIVERSITY LINZ, 2018.

[11] Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Mohammad Norouzi, Douglas Eck, and Karen Simonyan. Neural audio synthesis of musical notes with wavenet autoencoders. In *International Conference on Machine Learning*, pages 1068–1077. PMLR, 2017.

[12] Sebastian Flossmann, Werner Goebl, and Gerhard Widmer. The magaloff corpus: An empirical error study. *Proceedings of the 11th ICMPC. Seattle, Washington, USA*, 2010.

[13] Francesco Foscarin, Andrew Mcleod, Philippe Rigaux, Florent Jacquemard, and Masahiko Sakai. Asap: a dataset of aligned scores and performances for piano transcription. In *ISMIR 2020-21st International Society for Music Information Retrieval*, 2020.

[14] Anders Friberg, Roberto Bresin, and Johan Sundberg. Overview of the kth rule system for musical performance. *Advances in Cognitive Psychology*, 2(2-3):145–161, 2006.

[15] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[16] Maarten Grachten et al. *Expressivity-aware tempo transformations of music performances using case based reasoning*. Universitat Pompeu Fabra, 2006.

[17] Curtis Hawthorne, Erich Elsen, Jialin Song, Adam Roberts, Ian Simon, Colin Raffel, Jesse Engel, Sageev Oore, and Douglas Eck. Onsets and frames: Dual-objective piano transcription. *arXiv preprint arXiv:1710.11153*, 2017.

[18] Curtis Hawthorne, Andriy Stasyuk, Adam Roberts, Ian Simon, Cheng-Zhi Anna Huang, Sander Dieleman, Erich Elsen, Jesse Engel, and Douglas Eck. Enabling factorized piano music modeling and generation with the maestro dataset. *arXiv preprint arXiv:1810.12247*, 2018.

[19] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Ian Simon, Curtis Hawthorne, Andrew M Dai, Matthew D Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer. *arXiv preprint arXiv:1809.04281*, 2018.

[20] Dasaem Jeong, Taegyun Kwon, and Juhan Nam. Virtuosonet: A hierarchical attention rnn for generating expressive piano performance from music score. In *NeurIPS 2018 Workshop on Machine Learning for Creativity and Design*, 2018.

[21] Dasaem Jeong, Taegyun Kwon, Yoojin Kim, Kyogu Lee, and Juhan Nam. Virtuosonet: A hierarchical rnn-based system for modeling expressive piano performance. In *ISMIR*, pages 908–915, 2019.

[22] Dasaem Jeong, Taegyun Kwon, Yoojin Kim, and Juhan Nam. Graph neural network for music score data and modeling expressive piano performance. In *International Conference on Machine Learning*, pages 3060–3070, 2019.

[23] Dasaem Jeong, Taegyun Kwon, Yoojin Kim, and Juhan Nam. Score and performance

features for rendering expressive music performances. In *Proc. of Music Encoding Conf*, 2019.

[24] Shulei Ji, Jing Luo, and Xinyu Yang. A comprehensive survey on deep music generation: Multi-level representations, algorithms, evaluations, and future directions. *arXiv preprint arXiv:2011.06801*, 2020.

[25] Anssi Klapuri. Introduction to music transcription. In *Signal processing methods for music transcription*, pages 3–20. Springer, 2006.

[26] Eita Nakamura, Kazuyoshi Yoshii, and Haruhiro Katayose. Performance error detection and post-processing for fast and accurate symbolic music alignment. In *ISMIR*, pages 347–353, 2017.

[27] neptune.ai. Neptune: experiment management and collaboration tool, 2020. URL https://neptune.ai.

[28] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[29] Sageev Oore, Ian Simon, Sander Dieleman, Douglas Eck, and Karen Simonyan. This time with feeling: Learning expressive musical performance. *Neural Computing and Applications*, 32(4):955–967, 2020.

[30] Christine McLeavey Payne. Musenet, Nov 2020. URL https://openai.com/blog/musenet/.

[31] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

[32] Orjan Sandred, Mikael Laurson, and Mika Kuuskankare. Revisiting the illiac suite–a rule-based approach to stochastic processes. *Sonic Ideas/Ideas Sonicas*, 2:42–46, 2009.

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30:5998–6008, 2017.

[34] Gerhard Widmer. Machine discoveries: A few simple, robust local expression principles. *Journal of New Music Research*, 31(1):37–50, 2002.

[35] Gerhard Widmer. Getting closer to the essence of music: The con espressione manifesto. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(2):1–13, 2016.