Jazz Piano Lead Sheet Arrangement using Deep Learning

Elliot Hogg
200817750

David Herbert
Mark Turner
Jim Mcgrath

DD/mm/yyyy

15000 Words

Declaration

I hereby declare that this dissertation represents my own work except where otherwise stated.

Abstract

Over the past 10 years, deep learning has been extensively researched and used in the task of music generation. However, one area that remains mostly untouched, is the task of lead sheet arrangement. In the genre of Jazz, most music is represented in the lead sheet format – a single notated melody and chord symbols. The interpretation and arrangement of the music must be decided and undertaken by the performer. To tackle this problem, a novel conditional generative adversarial network (C-GAN) was designed to generate jazz chords which could be used to create full arrangements. Participant lead evaluations showed that resulting arrangements were found to be more effective than template chord selection.

Acknowledgments

I would firstly like to thank my supervisor David Hebert, whose support and advice was invaluable to this project.

I would also like to thank Mark Turner for his continued help and support throughout the project.

Contents

1. Introduction

Lead sheet arrangement is a task whereby a musician takes an abbreviated representation of a song, and adds harmonic and rhymical elements to transform it into a compelling piece of music. In the area of Jazz music, the lead sheet is the traditional and most common way for songs and pieces of music to be notated, mainly due to the fact that they allow for creative freedom, and leave much to the interpretation and direction of the musician, an idea which is essential to Jazz music.

Computationally, lead sheet arrangement can be defined as a process that takes a lead sheet as input, and outputs a harmonically and rhythmically complete digital representation of that song.

There are a few commercial applications that perform the harmonic element of lead sheet arrangement; however, they are inadequate, and simply select chords from a dictionary of chord voicings. The chord voicings themselves are generic and lack musicality.

The process of lead sheet arrangement can be defined as a sub task of music composition. If music composition is described as a three-step process, step one being creating a melody, and step 2 being choosing some chords to accompany that melody, then lead sheet arrangement is step 3.

(smoothen transition?)

And the focus on music composition within the field of deep learning has
has regained a large amount of academic and public attention within the past 10 years (citation). This is primarily due to the development of deep neural networks (Ji et al., 2020: 5). One great example of such generation is *DeepBach*, a deep learning model that can compose novel, convincing chorales in the style of composer Bach (Hadjeres et al., 2017). Deep learning has also been used to generate monophonic and polyphonic music of many different genres (citations needed).

Research and implementation of deep learning in the task of lead sheet arrangement is however very limited. There exist only two publications on the topic, both from the same authors. In their research, they employed deep learning to arrange pop lead sheets into a full arrangement for a 4 piece band (reference).

Music generation tasks are usually achieved using either recurrent neural networks (RNN) or generative adversarial networks (GAN). The former type of network is highly effective at generating temporal sequences, as the networks outputs are dependent on previous outputs (citation). The latter network is effective at generating completely new instances of data that are undistinguishable from a given set of data (citation).

The inherent nature of the lead sheet is that, once a musician has composed an arrangement, they tend to memorise it rather than notate it (citation). This is a particular issue in jazz, in which there exist very few arrangements of jazz standards in relation to the size of the total corpus (citation). For intermediate or amateur jazz musicians, or for advanced classically trained musicians, the task of lead sheet arrangement can be challenging, and can create an effective barrier to entry for the genre as a whole (reference).

Harnessing the power of deep learning in order to create a system capable of generating compelling arrangements would be invaluable to the jazz community, as it would make jazz music much more accessible.

The focus of this project is to create a lead sheet arrangement system that will take a lead sheet as input and output an arrangement. The system will also be intended to provide a framework for others to employ and use in order to advance the field of research. The system will be limited in that it will only address the harmonic element of the process, ie. It will provide chord voicings for the chord symbols, and not consist of any rhymic elements.

The system will consist of three key elements. (1) A chord scraper that is capable of extracting chord voicings from fully arranged songs with chord symbols. This scraper will be limited to songs in MusicXML format, which is a popular XML based music encoding language (reference). (2) A chord generator that will use a conditional generative adversarial network to generate musically pleasing and convincing chord voicings. The network will be trained using a novel dataset of labelled chords gathered by using the chord scraper on existing jazz piano arrangements. (3) A lead sheet arranger, which will take the melody from the lead sheet and the chords from the generator, combine them, and output an arrangement in MusicXML, MIDI, and Audio formats.

The project will also present the novel dataset of labelled chord voicings for use in further deep learning tasks.

1.2 Aim

To create a deep learning led system that can generate arrangements of any given jazz lead sheet.

1.3 Objectives

- To develop a program that can scrape chord voicings from fully arranged pieces of piano music with chord symbols in MusicXML format
- To present a novel dataset of labelled jazz piano chord voicings
- To create a novel conditional generative adversarial network that can output compelling colourings and voicings for chord symbols from a given lead sheet
- To develop a program that can combine chord voicings with the melody of a lead sheet in a way that is both convincing and playable.
- To perform some empirical and participant led testing on the outcomes.

1.4 Outline

The structure of the report follows both the order it was developed in as well as the order in which the system as a whole operates.

- **Chapter 2** provides a review of the recent literature within the field of deep learning as it pertains to music. There will also be a critical evaluation of existing lead sheet arrangement models.
- **Chapter 3** provides information about the data used in the project.
- **Chapter 4** provides a deep dive into how the three key components of the system were conceptualised and developed. Starting with the chord scraper, then the C-GAN deep learning model, and finally the lead sheet arranger.
- **Chapter 5** presents an evaluation of each component, as well as an evaluation of the results of the system as a whole. Evaluations of the system are both technical and participant led.
- **Chapter 6** presents a conclusion of the project, with commentary on the how initial aims and objectives were met, as well as some future considerations and recommendations for further research.

2. Background Research

2.1. Literature review of deep learning in music

2.2. Critical evaluation of useable models

2.2.1 Lead sheet generator and arranger

2.2.2 Pix2Pix GAN

2.3. Survey of available datasets

2.4. Identifying Technologies

Paper	Year Of Pu...	Author(s)	ML Models
Generating Lead Sheets with Affect: A Novel Conditional seq2seq Framework	May 2021	Dimos Makris, Kat R Agres, Dorien Herremans	Transformer Network, Long Short-Term Memory (LSTM)
Automatic melody harmonization with triad chords: A comparative study	April 2021	Yin-Cheng Yeh, Wen-Yi Hsiao, Satoru Fukayama, Tetsuro Kitahara, Hao-Min Liu, Yi-Hsuan Yang, Benjamin Genchel, Hao-Wen Dong, Yian Chen, Terence Leong	Bidirectional long short-term memory network (biLSTM)
Generative Deep Learning for Virtuosic Classical Music: Generative Adversarial Networks as Renowned Composers	January 2021	Daniel Szelogowski	Support Vector Machine (SVM), Generative Adversarial Network (GAN), Recurrent Neural Network (RNN), Attention, Long Short-Term Memory (LSTM)
Vertical-Horizontal Structured Attention for Generating Music with Chords	November 2020	Yizhou Zhao, Liang Qiu, Wensi Ai, Feng Shi, Song-Chun Zhu	Recurrent Neural Network (RNN)
A Comprehensive Survey on Deep Music Generation: Multi-level Representations, Algorithms, Evaluations, and Future Directions	November 2020	Shulei Ji, Jing Luo, Xinyu Yang	N/A, Literature Review
CHORD JAZZIFICATION: LEARNING JAZZ INTERPRETATIONS OF CHORD SYMBOLS	October 2020	Tsung-Ping Chen, Satoru Fukayama, Masataka Goto, Li Su	Multihead Self-attention Network (MHSA), Cross-Validation, Long Short-Term Memory (LSTM), Bidirectional recurrent neural network (BRNN)
POP909: A POP-SONG DATASET FOR MUSIC ARRANGEMENT GENERATION	August 2020	Ziyu Wang, Ke Chen, Junyan Jiang, Yiyi Zhang, Maoran Xu, Shuqi Dai, Guxian Bin, Gus Xia	N/A, Literature Review
THE JAZZ TRANSFORMER ON THE FRONT LINE: EXPLORING THE SHORTCOMINGS OF AI-COMPOSED MUSIC THROUGH QUANTITATIVE MEASURES	August 2020	Shih-Lun Wu, Yi-Hsuan Yang	N/A, Literature Review
Pop Music Generation: From Melody to Multi-style Arrangement	August 2020	Hongyuan Zhu, Qi Liu, Nicholas Jing Yuan, Kun Zhang, Guang Zhou, Enhong Chen	Recurrent Neural Network (RNN)
AN LSTM-BASED DYNAMIC CHORD PROGRESSION GENERATION SYSTEM FOR INTERACTIVE MUSIC PERFORMANCE	May 2020	Christos Garoufis, Nancy Zlatintsi, Petros Maragos	Long Short-Term Memory (LSTM)
Dual-track Music Generation using Deep Learning	May 2020	Sudi Lyu, Anxiang Zhyang, Rong Song	Long Short-Term Memory (LSTM), Convolutional Neural Network (CNN), Attention
AN INTERACTIVE WORKFLOW FOR GENERATING CHORD LABELS FOR HOMORHYTHMIC MUSIC IN SYMBOLIC FORMATS	November 2019	Yaolong Ju, Samuel Howes, Cory McKay, Nathaniel Condit-Schultz, Jorge Calvo-Zaragoza, Ichiro Fujinaga	Support Vector Machine (SVM), Deep Neural Network (DNN)
MIDI-Sandwich2: RNN-based Hierarchical Multi-modal Fusion Generation VAE networks for multi-track symbolic music generation	September 2019	Xia Liang, Junmin Wu, Jing Cao	Recurrent Neural Network (RNN)
Lead Sheet Generation and Arrangement by Conditional Generative Adversarial Network	December 2018	Hao-Min Liu, Yi-Hsuan Yang	Generative Adversarial Network (GAN)
Machine learning research that matters for music creation: A case study	September 2018	Bob L. Sturm, Oded Ben-Tal, Úna Monaghan, Nick Collins, Dorien Herremans, Elaine Chew, Gaëtan Hadjeres, Emmanuel Deruty, François Pachet	N/A, Literature Review
CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORKS WITH BINARY NEURONS FOR POLYPHONIC MUSIC GENERATION	September 2018	Hao-Wen Dong, Yi-Hsuan Yang	Convolutional Neural Network (CNN)
LEAD SHEET GENERATION AND ARRANGEMENT VIA A HYBRID GENERATIVE MODEL	September 2018	Hao-Min Liu, Meng-Hsuan Wu, Yi-Hsuan Yang	Generative Adversarial Network (GAN)
Project milestone: Generating music with Machine Learning	June 2018	David Kang, Jung Youn Kim, Simen Ringdahl	Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM)
CHORD GENERATION FROM SYMBOLIC MELODY USING BLSTM NETWORKS	December 2017	Lim Hyungui, Rhyu Seungyeon, Lee Kyogu	Bidirectional long short-term memory network (biLSTM)
JamBot: Music Theory Aware Chord Based Generation of Polyphonic Music with LSTMs	November 2017	Gino Brunner, Yuyi Wang, Roger Wattenhofer, Jonas Wiesendanger	Long Short-Term Memory (LSTM)
GENERATING NONTRIVIAL MELODIES FOR MUSIC AS A SERVICE	October 2017	Yifei Teng, An Zhao, Camille Goudeseune	Autoencoder, Recurrent Neural Network (RNN)
MUSEGAN: DEMONSTRATION OF A CONVOLUTIONAL GAN BASED MODEL FOR	October 2017	Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, Yi-Hsuan Yang	Generative Adversarial Network (GAN)

Classification model notes

Represented the data in 3 different ways and used RNN and CNN to run classification tasks

Report results…

Look at where label classifications were different that expected -> are they in any way more accurate? Could this potentially be used to classify chords without chord symbols in XML files?? To gather more data??

GAN notes

Embedding: transform indexes into a vector of fixed size. E.g., representing words as numbers

One-hot embedding: create a "sparse" binary vector that is the length of the dictionary (100 words = 100 length vector). Then each word is represented by a 1 instead of a 0 in the vector.

____

https://www.youtube.com/watch?v=BUNl0To1IVw&t=3018s:

Generative models:

Autoencoders and Variational Autoencoders (VAEs):

Learn lower-dimentional **latent space** and **sample** to generate input reconstructions

GANs:

Don’t explicitly model density, and instead just sample to generate new instances

When generator is trained – it is learning to transform from distribution of noise to target data disttibution

Gaussian noise → target data manifold
Z → Y

Take on point from latent noise distribution – will result in particular output in target data space.

The points in latent noise distribution represent different outputs

If we transverse in the Gaussian noise space we get interpellation of output (2 outputs mixed together)

55:10 – conditional GAN

Introduce pairs to G and D

Paired translation

12 vector integer notation -> 88 vector full representation

Effective distribution transformers

____

How c_gan works exactly:

The generator takes in a label and outputs a prediction

Label -> **Generator** - > Fake Chord

Label + Fake Chord || Real Chord -> **Discriminator** - > Real or Fake

The discriminator takes in pairs of labels + fake / real occurences and outputs a value on whether or not they are real

After the model is trained we give the generator labels and it gives us predictions

Adapted pix2pix model:

Transformed chord notes and label data into 7x12 matrices (omitting bottom 3 notes and top note)

Adapted pix2pix model to contain 1D layers as oppose to 2D layers (see paper)\

Initial results:

The loss of the discriminator on both real and generated data rapidly went to near zero, why? Probably the generator

Novel model

“Both G and D could be a non-linear mapping function, such as a multi-layer perceptron.”

Discriminator

Discriminator that takes in concatenation of labels and either fake/real chords

How should this data be represented/concatenated?

Look at results of classification tasks
When concatenating label and chord, should an extra dimension be added?

Discriminator should have a single output with sigmoid activation (value between 0 and 1 indicating whether or not it is fake/real)

If using 1d representation, should concatenation be on a third axis, i.e. (88,) + (88,) → (1, 176) **OR** on the x axis, ie. (88,) + (88,) → (176,)

How can this decision be made? Other than just testing it, is there some reasons for why one would perform better than the other?

Probably the third axis, as it represents the relationship between label and chord in a more meaningful way – EXPAND ON THIS

If using 2d representation of labels/chords -> what is the use of convolutions? Can they actually find patterns -> research this

Generator

Takes in a label and transforms it into a fake chord candidate

Again, do we need to use convolutional layers? Or will deep/fully connected layers do the job

Real examples = (label + real chord) + [1] (label for discriminator)

Fake examples = (label + fake chord) + [0] (label for discriminator)

Training

GAN Loss Function:

the generator tries to minimize the following function while the discriminator tries to maximize it:

$$\min_G \max_D E_X \left[\log(D(x))\right] + E_z \left[\log\left(1 - D(G(z))\right)\right]$$

- D(x) is the discriminator's estimate of the probability that real data instance x is real.
- E_x is the expected value over all real data instances.
- G(z) is the generator's output when given noise z.
- D(G(z)) is the discriminator's estimate of the probability that a fake instance is real.
- E_z is the expected value over all random inputs to the generator (in effect, the expected value over all generated fake instances G(z)).
- The formula derives from the cross-entropy between the real and generated distributions.

(https://developers.google.com/machine-learning/gan/loss)

Conditional GAN Loss Function

$$\min_G \max_D E_X \left[\log(D(x|y))\right] + E_z \left[\log\left(1 - D(G(z|y))\right)\right]$$

Backpropagation

Generator

Results:

Data Embedding

I experimented with the data embedding:

1) 2 different 1D representations:

1. Source chord, real chord = (12,)(88,)

2. Source chord, real chord = (88,)(88,) [source chord(label) embedded to full 88 note vector in the middle C octave

2) a 2D representation:

(88,)(88,) 1D embeddings reduced to (84,)(84,) by removing bottom 3 notes and top note. Then transformed to (7,12)(7,12)

Classification Models

In order to determine the best embedding for the GAN, as well as the best type of network for the discriminator and generator, performed some classification tasks using all the embeddings and a combination of RNN and CNN.

1D RNN classifier using both **1.1 and 1.2** (above)

2D CNN classifier using **2**

Results:

Both performed incredibly well:

Give functions, data, graphs, etc

Model 1: 2d pix2pix model (adapted for my dataset):

Results showed that the discriminators loss quickly went to near zero – meaning that the generator was not generating convincing chords from source chords.

Show graph of loss over time and any other graphs/chards that were in 1D paper or other papers

Show some example generated chords

Why?

Unclear, using convolutional layers on simple binary chord matrices potentially didn't yield any meaningful patterns

patchGAN uses a segment of the real/fake images (chords) passed in. ie. If matrices is 128x128, it looks at a 16x16 segment or multiple segments

this approach wont be as effective for chord data, as there would be notes spread around the vector, particularly on the x-axis (0 axis??)

Also mention loss functions /learning rate etc

Model 2: 1D pix2pix model (adapted for my 1D dataset):

Results?

Same as above

Model 3: 1D Novel Model

This model was born out of some initial testing using novel classifier networks

V1

The discriminator had a single output neuron with sigmoid activation with binary-crossentropy loss function

Results…. Loss of the discriminator on fake chords quickly reduced to 0, meaning generator was not improving *mention the loss function of the GAN here* - explain it etc

more results here with graphs etc and chords shown in images etc with stats

V2

Created an 88 neuron output layer on the discriminator

* mention how this changes the way the loss function works in the c_gan model

Discriminator loss on real chords went from 1.5 – 1.2 over 11 thousand iterations of training
Loss on fake chords quickly went to zero

Changed way source chord and real/fake chord concatenated

Initially it was (88,)(88,) → (2, 88)
Maybe dense layers don't perform as well on matrices?? Reference/research

Changed to (88,)(88,)

Leadsheet XML Arranger

INPUT: leadsheet (musicxml)

Step 1. Extract chord symbols
Step 2. Embed chord symbols as 88 "source-chord" note vectors (see preprocessing chapter)
Step 3. Pass source chord vectors into c_gan model
Step 4. Combine leadsheet melody with generated chords into full arrangement

OUTPUT: Chords + Melody Arrangement

Data parsed from lead sheets

Meta information:
-Key
-Time signature

Chords:

-Chord symbol
-Bar number
-Position in bar
-Note at chord position (won't account for 2 voicings / not sure which note it will get if chord present)

<u>Data Issues</u>

The number of different types of chords in the data was skewed :

Insert pandas table

1000 dominant 7 wheras much less of other chords

Data skewness

Data should be bell curve

But currently it is right skewed

Something to do with mode median

SOLUTION?

Get more data + remove some label occurences

WHEN SPLITTING TEST vs TRAIN:

<u>INITIAL DATA COUNTS WITHOUT WIDENING CHORD SELECTIONS:</u>

```
dominant              1177
minor-seventh         1053
major-seventh          674
major                  453
dominant-ninth         255
suspended-fourth       249
dominant-13th          159
minor-ninth            133
half-diminished        129
minor                  117
major-sixth             96
major-ninth             68
minor-11th              63
diminished-seventh      59
diminished              44
```

```
minor-sixth             38
Name: label, dtype: int64
```

Performing data analysis using pandas and matlabplot

Removed all chord labels that have less than 600 occurrences

Created a stacked bar chart representation of each label to analyse data and note occurrences

insert bar charts

For the large part the data was good - **insert** margin of error numbers

However a small number of labels had wrong notes associated with them

insert some examples with numbers of wrong notes

Therefore further data cleaning was performed. Created array of unwanted notes for each label, and iteratively removed those notes from each chord vector.

** insert cleaned bar charts**

We now have

References

Ji, S., Luo, J. and Yang, X., 2020. A Comprehensive Survey on Deep Music Generation: Multi-level Representations, Algorithms, Evaluations, and Future Directions. arXiv preprint arXiv:2011.06801.

Hadjeres, G., Pachet, F. and Nielsen, F., 2017, July. Deepbach: a steerable model for bach chorales generation. In International Conference on Machine Learning (pp. 1362-1371). PMLR.

Mirza, M., & Osindero, S. (2014). Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784.