Improving Chord Prediction in Jazz Music

Nikunj Sangwan[1], Oskar Hallström[2], and Anmol Prasad[2]

[1]Department of Physics, EPFL
[2]Department of Computer Science, EPFL

December 23, 2021

Abstract

This project aims to predict the next chord of a jazz performance at any given time of the performance, and then to see if these predictions can be improved by using melody information. We train LSTMs using chord and melody data from the Weimar jazz database, which contains 456 performances. For our baseline model which only uses chord information, we obtain a test accuracy of about 50%. Feeding the melody as a bag of notes into our neural network, our test accuracy increases to about 51%.

Keywords ML, jazz, chord prediction, LSTM

1 Introduction

Sequential chord prediction is the task of predicting the next chord given a chord progression, where chord progression simply refers to the order in which the chords are played. Sequential chord prediction has relevant applications in musical processing applications and music-theoretical investigations [1].

The main objective of this project is sequential chord prediction and investigating whether the prediction performances improve taking the melody information into account. In order to do that, two models were studied, a model that takes melody information into account and a baseline model which does not.

Predicting future chords from a given sequence is similar to language modelling where the main goal is to predict the next word in a sentence given the previous words. There have been many works on the sequential chord prediction. Earlier works use rule based models or probabilistic HMMs[2] but these do not capture the long-term dependencies very well. Similarly, N-gram models have also been used[3]. To improve on these, there has been more work on using RNNs and LSTM based methods. Cunha et Al [4] proposed a hybrid model, combining probabilistic methods with a neural network predictor. More recently, there has been more experimentation with RNNs, GRUs and LSTMs. Korzeniowsk et Al [3] show that compared to N-gram models, RNNs are much better in terms of capturing the long term temporal information and can adapt to the songs at test time. Therefore, we use LSTMs to predict the next chord in a sequence of chords. We also investigate whether the predictions are more accurate for chords that are simple and less accurate for complex occurring chords, like it would be the case for a human.

The Weimar jazz database (WJAZZD)[5] is used as the data set in this study . WJAZZD is collection of 456 jazz performances with the information about the beats and melody for each of the performances. We use this dataset because it has a large collection of clean jazz solo transcriptions of high quality and multiple styles. As the pitch classes of previous chords can help us predict the next chord, we hypothesise that the pitch classes of the melody could provide additional information of the future chords. In the rest of the paper, we will provide more information about the database[2.1] used, the pre-processing and encoding of the data[2.2] to be used for LSTM, the methods used to take information from melodies[2.3] into account and the results[4] achieved followed by discussion and possible future work.

2 Data and preprocessing

2.1 The Database

The WJAZZD database comprises of 456 jazz performances in total. The data is distributed into different tables. The "beats" table is the table for beat annotation and the chord values of WJAZZD melodies, referenced by melody (melid) and "melody" table is the main table for all melody events. There are other tables in the database which we did not use. Interested users can go to the link[1] to check information on other tables.

Field	Type	Description
beatid	INTEGER	Unique ID of beat event
melid	INTEGER	References melody(melid)
chord	TEXT	Accompanying chord
onset	REAL	Onset (in secs) of beat

Table 1: Overview of the beats table.

Each chord is represented in standard string format: first a root note (e.g., C, Bb), followed by a string or sign representing the quality of the chord, followed by the extensions, which are numbers which could be coupled with a sharp or a flat sign. If a bass note is specified, it is represented in the end of the string with a slash and then a pitch specifying the pitch class of the bass note. For instance, a G minor chord with F as the bass note is represented as G-/F, and a C dominant-seventh chord extended with a flat 9 is represented as C79b.

The melody is represented as a sequence of MIDI notes using an integer notation to encode the pitch class and octave. We map this to MIDI pitch class by taking the MIDI note number modulo 12

Field	Type	Description
melid	INTEGER	Unique ID of the melody containing the event
pitch	REAL	Pitch (fractional MIDI) of the event
onset	REAL	Onset (in secs) of beat
duration	REAL	Duration (in sec) of the event
beat	INTEGER	Beat number of the event

Table 2: Overview of the melody table.

In Tables 1 and 2 we show the columns used for our work.

Each performance is represented by a "melid" which identifies the jazz performance in consideration. A sequence of chords is associated with each "melid".

2.2 Mapping of chords

All the chord types of the data set are projected onto a reduced set of chord types. This is done to reduce sparsity. The bass note of a chord is always removed. The reduced set of chord types consists of essential chord types and chord types of which the occurrences correspond to at least one percent of total occurrences of chord types. The essential chord types are considered to be the non-extended triad version of all chord types represented in the original data set. These are the minor, major, diminished, augmented and suspended chords.

Essential chord types	Chord types of reduced set
major	major, 6, 7, 79b, maj7
minor	m, m6, m7, m7b5
augmented	aug
diminished	dim
suspended	sus, sus7

Table 3: Overview of used chord types.

If a chord is not in the reduced set, it is reduced iteratively by removing the highest extension note of the chord until it matches a chord in the reduced set. If the reduced set would only consist of a standard major chord, C79b would first be reduced to C7, then C, where the reduction would stop since the chord type now matches a chord type in the reduced set. For example, the following projections has been carried out: 79#13 to 7, maj7911#13 to maj7 and augmaj7 to aug.

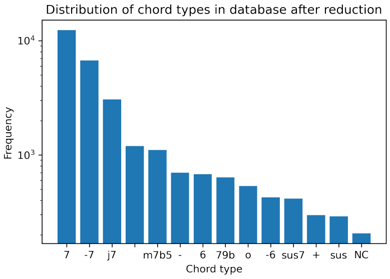

Figure 1: Chord type distribution in data set after reduction, with the y-axis on a log scale.

We represent a reduced chord with a multi-hot vector with 24 elements, comprising of 2 parts, the chord type and the root. The first 12 elements corresponds to the chord type, whereas the last 12 elements encode the root. The representation is based on the integer notation of pitch classes, in which C corresponds to 0, C# or Db to 1 and so on until B which corresponds to 11. For the last 12 elements, the root is indicated with a 1 at the index which matches its pitch class integer notation.

The chord type is represented through viewing it as a chord with root C, and then setting the elements at indices matching the pitch classes contained in the chord to 1. However, index 0, which corresponds to the root, is always set to 0. For a regular major chord, we consequently set the elements at index 4 and 7 to 1, and the rest of the first twelve elements to 0.

2.3 Melody mappings

For using the melody data we use the beat onset values to filter out the notes which appear in

[1] https://jazzomat.hfm-weimar.de/dbformat/dbformat.html

Chord	Multi-hot vector
C	[0,0,0,0,1,0,0,1,0,0,0,0, 1,0,0,0,0,0,0,0,0,0,0,0]
D7	[0,0,0,0,1,0,0,1,0,0,1,0, 0,0,1,0,0,0,0,0,0,0,0,0]

Table 4: Example of chord representations.

the time period when one chord is being played. These notes are denoted by the pitch values for which we store a normalized count vector of length 12. We refer to this vector as the Bag of Notes.

3 Models

3.1 Set ups

We use a LSTM based model. The encoded input sequence is first passed through an LSTM layer. The output is then passed through a fully connected layer after dropout.

We create 2 models with differing inputs: Chord-Sequences which is referred to as baseline, Chord-Sequences+Bag of notes, which is referred to as the Melody model. The input encoding is as follows:

- Chord-Sequence: For the baseline model, we encode each jazz performance as a series of encoded chords. We use the chord representation explained in 2.2 to encode the chords.

- Chord-Sequence + Bag of notes: In the case of the melody model, the melody mappings 2.3 for each chord are then concatenated to the multi-hot encoding to create a sequence of input vectors of the form chord_representation+melody_information

The dataset is divided into train, test and validation with an 80%,10%,10% split.

3.2 Training

The vocabulary size for the model is the number of possible outputs (157 in our case).

Grid search over different hyper-parameters were implemented over the values (0.1, 0.2, 0.3) for dropout size, (300, 350, 400) for hidden layer size, (1e-3, 1e-2, 1e-1) for the learning rate and (1e-5, 1e-4 and 1e-3) for the weight decay of ADAM optimiser to select the best hyper-parameters.

After grid search, a learning rate of 0.01 was selected because it prevented blowing up the losses over epochs and the reducing of learning rate on plateau made the program flexible enough to change the learning rate if the learning stagnates. All the models were trained over 400 epochs with an LR of 0.01. Early stopping with a patience of 20 and ReduceLROnPlateau[2] with a patience of 10 was used.

The results obtained by the grid-search on the hyper-parameters are presented in the results section.

4 Results and Discussion

4.1 Quantitative

4.1.1 Grid search

For the "baseline" model, best results were achieved with the hyper-parameters: dropout probability as 0.3, weight decay as 1e-4, hidden layer size as 350, while on the other hand for the "melody" model best results were achieved with dropout probability of 0.1, weight decay of 1e-4, hidden layer size being 400.

4.1.2 Testing

We test the model over a held out test set, resulting in accuracy measures given in Table 5.

	Baseline model	Melody model
Test accuracy	50 %	51 %
SEM of accuracy	4.4 %	4.3 %

Table 5: Overview of model accuracy.

4.1.3 Distributions of incorrect predictions

For the baseline model, 77.1% of the incorrect predictions contained the wrong root note. For the melody model, the corresponding percentage was 78.6 %. The most common of these errors can be seen in Table 6. The most common errors of predictions with the correct root can be seen in Table 7.

Target type	Predicted type	Root difference (predicted - target)	Frequency (of all errors)
		Baseline	
7	7	6	5.5 %
m7	7	7	2.9 %
7	7	2	2.9 %
major	major	7	2.6 %
major	major	4	2.3 %
		Melody	
7	7	6	6.7 %
major	major	2	3.3 %
major	major	7	2.2 %

Table 6: Overview of most common errors when root was wrong.

[2] https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.ReduceLROnPlateau.html

Target type	Predicted type	Frequency in baseline (of all errors)	Frequency in melody (of all errors)
79b	7	3.6 %	3.5 %
m7	7	2.8 %	3.3 %
maj7	7	2.1 %	1.6 %
7	m7	2.1 %	1.5 %

Table 7: Overview of most common errors when root was right.

4.1.4 Discussion of quantitative results

Both models almost have the same test accuracy. The melody model and the baseline model make errors in similar ways too. The proportion of predictions which are wrong for the chord type, but right for the root, is similar. The four most common errors in this case are identical for both models. The most common error is to predict a dominant seventh chord when the actual chord is a dominant seventh flat ninth chord. This error is likely due to dominant seventh flat ninth chords being less frequent then dominant seventh chords, which is the most common chord type in the data set after reduction. The fact that the dominant seventh chord is the most frequent could also explain the second and the third most common error, which is to predict a dominant seventh chord instead of a minor seventh chord and a major seventh chord respectively. The fourth most common error is to predict a minor seventh chord instead of a dominant seventh. Even though dominant seventh chords are the most common, minor seventh chords are the second most common ones, which makes this error reasonable in terms of previous frequency reasoning.

When it comes to false predictions in which the root is wrong, both model's most common error is to predict a dominant-seventh chord, when the target is a dominant seventh chord that is transposed by 6 semitones in comparison. The chords of this tritone confusion share two of 4 pitches, and fulfill the same harmonic function in jazz. This error is thus reasonable to appear, and something a real musician likely could do. So despite both model predicting the wrong chord in about half of the cases, many of the errors are not necessarily that far from the actual chord.

4.2 Qualitative

In Figure 2, a prediction sample is given for the first song in the test set.

Despite similarities, the actual predictions can often differ between the models, as seen from in the given sample. In the 9 cases where the predictions differ between the models, the melody model's prediction shares more pitches with ground truth in 7 of them. The Wow sample could thus imply that the melody model's incorrect predictions are better than those of the baseline model. For instance, the melody model predicts Gm in beat 9 where the actual chord is Gm7, while the baseline prediction is Em7.

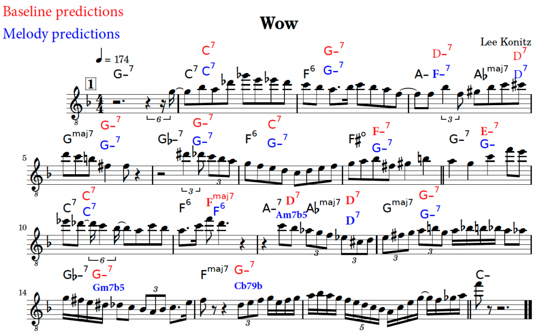

Figure 2: Predictions on Wow of the two models.

5 Conclusions

In this paper, we presented a model for chord prediction in Jazz. Our results showed that a model which uses melody information did not outperform a baseline which used only chord symbols in terms of accuracy. Since our results are not significant enough to verify if melody can improve prediction performance, this is thus a research question that would need future investigation.

From the discussion, it appears that the evaluation of the models can not be solely based on accuracy, since this measure does not capture the quality of the errors. A significant part of the errors made by both models are not severe from a theoretical perspective. In addition, the qualitative analysis could indicate that quality of errors generally is better for the melody model. To be able to capture this on a larger scale, future work could benefit by using a broader definition of prediction performance. This by implementing a more rigorous measure of error quality, such as percentage of pitches shared by predicted chord and ground truth.

For future work, we also suggest a more advanced implementation of melody information, such as for instance weighting the notes differently. This weighting could be made based on for instance pitch duration or the order of pitches.

Acknowledgements

Thanks to our supervisors Daniel Harasim and Andrew McLeod from the Digital and Cognitive Musicology Lab at EPFL for their valuable support and help.

References

[1] Allison Lahnala, Gauri Kambhatla, Jiajun Peng, Matthew Whitehead, Gillian Minnehan, Eric Guldan, Jonathan K. Kummerfeld, Anil Çamci, and Rada Mihalcea. Chord embeddings: Analyzing what they capture and their role for next chord prediction and artist attribute prediction. *CoRR*, abs/2102.02917, 2021.

[2] Jean-Francois Paiement, Douglas Eck, and Samy Bengio. A probabilistic model for chord progressions. pages 312–319, 01 2005.

[3] Filip Korzeniowski, David R. W. Sears, and Gerhard Widmer. A large-scale study of language models for chord prediction. *CoRR*, abs/1804.01849, 2018.

[4] Uraquitan Sidney Cunha and Geber Ramalho. An intelligent hybrid model for chord prediction. *Organised Sound*, 4:115–119, 1999.

[5] Martin Pfleiderer, Klaus Frieler, Jakob Abeßer, Wolf-Georg Zaddach, and Benjamin Burkhart, editors. *Inside the Jazzomat - New Perspectives for Jazz Research.* Schott Campus, 2017.