EXPLORING GENERATIVE MODELS FOR CREATING CHORD PROGRESSIONS

Jonathan Adam
KTH
ja11@kth.se

Adrian Latupeirissa
KTH
ablat@kth.se

ABSTRACT

We explore various approaches to generating chord progressions using methods involving generative models. With a dataset of compositions from Bach and his contemporaries, we train a series of hidden Markov models of various orders. We use these HMMs to harmonize an unfigured bass, both in real time (in collaboration with a player) or with knowledge of the full bass line. The voice leading is performed through an "expert system" algorithm based on music-theoretical constraints. We also attempt to train a LSTM recurrent neural network on the raw data, to see whether this strategy can simultaneously track harmonic and melodic issues.

1. BACKGROUND

A significant research area within computer music is creating programs which can compose music resembling that of a human composer. Creating these programs is both an exciting challenge in determining how convincing an algorithm can perform musical creativity, and simultaneously provides insight into what makes human creativity so particular.

There are three main approaches to training a system into composing human-like music. The first would be to create an "expert system," whereby one hard-codes the rules of a particular musical genre into a program. This approach may be well suited for incredibly rigorous musical systems, but has significant drawbacks. It is by definition inflexible and needs to have an appropriate rule set before it can accurately emulate its origins. For convincing results these rules would have to cover more than just explicit aspects of the system and include some kind of guidelines to favor "human" decisions. Nevertheless, expert systems remain a topic of interest in various aspects of music generation. Donya Quick's work focuses on encoding musical theoretical principles of form, such as those set forth by Schenkerian analysis, into formal grammars. [1] In doing so, computer music can generate larger-scale form.

Another main area of research is using genetic algorithms for music generation. [2] In these approaches, a large number of samples is created and evaluated in terms of a fitness function. Through processes modeling natural selection these samples work towards finding an optimal solution. The difficulty in developing these systems is developing an appropriate fitness function for musical structures. Furthermore, due to their long runtime they may not be the most suitable methods for systems aimed at real-time interaction.

The third approach, machine learning, enjoys a fair amount of popularity and success in the research field. Covering a vast set of approaches and specific strategies, the main goal of machine learning is to train a system not with explicit rules or optimisation functions, but implicitly, by allowing its internal organization to alter its parameters to fit to input data. Particularly Hidden Markov Models and Recursive Neural Networks have enjoyed successes in the field. [3] [4] Recent headlines have featured the first pop song to be written with the help of neural networks, from the same team who used the same network to generate Bach chorales. [5]

2. METHODS

While many research approaches attempt to create all-in-one solutions to music generation, we decided to focus on just harmony, specifically harmonizing bass lines, eliminating concerns about rhythm and (for the most part) melody. We decided to focus primarily on Hidden Markov Models, as these were a model that we already had previous encounters with and which had appeared regularly in the literature. However, where past research had used them primarily for melodic generation (also within the context of generating contrapuntal melodic strands to create harmony), we wanted to upend that hierarchy and derive our harmonies from the bass up. This conceptual model of music generation is related to the accompaniment of unfigured bass, where an instrumentalist gets a bass line with minimal to no further harmonic information, and has to extemporize a harmonic accompaniment.

2.1 Dataset and Preprocessing

The Python library music21 comes with a built-in corpus of various composers. [6]. We primarily used the works of Bach, and occasionally expanded this dataset to also include works of Buxtehude and other contemporaries to increase the training data. We also attempted collecting datasets from other composers, notably Chopin, but these were smaller and were less inclined to lend themselves well to our analysis.

We transposed all scores in the dataset to C major or A minor depending on their mode to align chord distributions. Next, we used music21's "chordify" function to collapse all the pitch-related musical events into a series of chord events. It is these chords that form the building blocks of our generative models.
For the HMM training we wrote out the chords into a bass note name followed by a pitch class representation of the chord, effectively eliminating any consideration of voicing and focusing exclusively on chord quality.
We used another representation for the purpose of the neural network training. As the RNN generates individual characters, we decided to see what would happen if we embed these characters with as much meaning as possible. While still omitting rhythmic information, we represented our dataset for the system as a series of words, where each letter corresponded to the character encoding of a note in the chord's MIDI pitch. This dataset consisted of approximately 0.5 Mb as a text file.

2.2 Hidden Markov Models

We began our investigation by modeling chord progressions as Markov chains. This model views every chord as a state, with a certain attached probability that it transitions to another state. We could easily generate chord progressions by performing a random walk over the states, sampling from the transition probability distributions at every step.
To train this model from our bass-chord data, we counted the number of times each bass note was harmonized by a certain chord, and how often each chord progressed to a subsequent chord. Normalizing these figures (i.e., dividing by the number of occurrences of the event) gives probabilities for each transition.
The characteristic feature of a Markov chain is that its current behavior is independent from all its past steps. This ensures an easy generation but complicates building in long-term dependencies and behavior. To counteract this tendency of the model to "wander" aimlessly, we trained a number of models on *n*-gram states, where *n* took values of 2 to 8. Concretely, instead of having each state represent one single chord, each state represented a chord appearing after a certain chordal progression of *n - 1* chords had preceded it. This gives each state an implicit sense of "memory" over the last few chords it has had to visit, and allows the model to make decisions conditioned on a number of preceding steps.

2.2.1 Real-time unfigured bass

Having trained an HMM on bass notes and harmonies, we can use it to generate suitable suggestions for chords based on an input note. Doing so simply requires finding the maximum hidden state probabilities for any next step in the process.

2.2.2 Finding the likeliest chord progression

Markov chains are good at modeling random processes, but we wanted to also investigate whether we could use them to model longer-term harmonic tension. For this, however, it only makes sense that the model has a sense of where the bass line is headed. Given a complete bass line, we implemented the Viterbi algorithm. This algorithm will find the series of hidden states that is most likely to explain the observed bass tones through a dynamic programming table.

2.3 Voice Leading

The Hidden Markov Model will return the pitch class set of the chords it deems appropriate for a certain bass note. However, these pitch classes do not contain any information about register and smooth voice leading between chords - an important stylistical feature of a proper unfigured bass realization. We decided to attempt to create a simple "expert system" that programmatically established how each chord should move to the next, following some simple rules:

- Common tones between the two chords should be held;
- Individual voices should move as little as possible;
- Upper voices should move in contrary or oblique motion relative to the bass as often as possible;
- The full harmony of the chord should be heard as often as possible.

These rules were gleaned from work done by Tymoczko. [7] We did not arrive at one of the canonical rules of the style - no parallel fifths or octaves - but avoided this behavior implicitly by favoring chords in inverted voicings and contrary motion to the bass.

2.4 Training a neural network

We wanted to compare our HMM-expert sytem hybrid with a solution tackling the same problem from a very different angle. We used a character-level LSTM recurrent neural network developed by Andrej Karpathy. [8] This network consists of a layer of "neurons" which fire at certain activations. What distinguishes recurrent neural networks from feed-forward networks is that decisions are made based on not just the input given, but also on the past internal states of the network. As such, it is capable of internally retaining a sort of memory - which should promote longer dependencies in the output and give overall senses of structure. We train this network on a text file of words where each character corresponds to a particular MIDI note. At regular iteration intervals the network retains its current training state and evaluates its performance based on a small subset of the data which it holds off for validation. It is not guaranteed that the longer a model trains the better it performs, as there is a risk of overfitting.

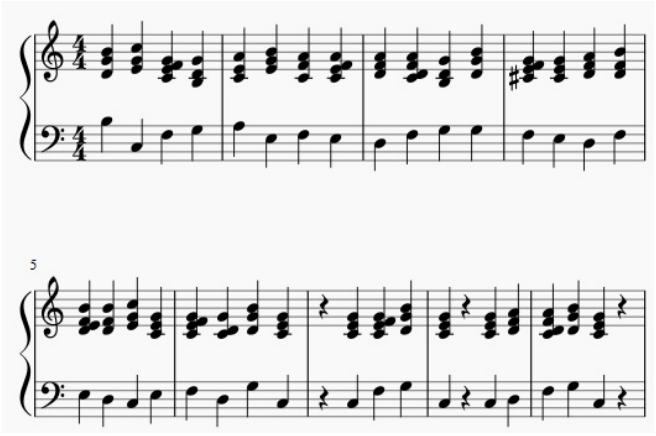

Figure 1. Output from the Viterbi harmonization with the voice leading expert sytem.

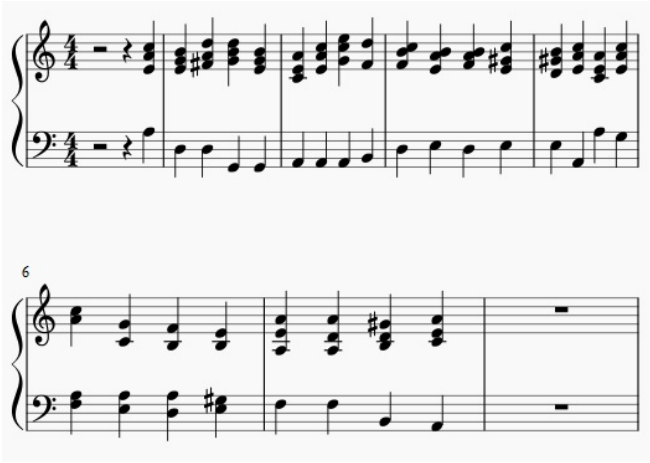

Figure 2. Sample output from the Neural network, when its loss function was at its minimum. Observe the minor V chord which seems erroneous, but an overall decent cadential shape.

3. RESULTS

3.1 Real-time chord suggestion

Our hidden Markov Models are able to succesfully suggest appropriate harmonies given consecutive bass notes. As soon as the model is done training it can instantaneously provide recommendations. In general, all n-gram models treat cadential passages well, with *ii-V-I* or *IV-V-I* cadences being strongly recommended by the system. When working with larger n-gram we notice that the system becomes more particular in its judgements, an indication that it might be literally quoting from the corpus.

3.2 The Viterbi algorithm

The Viterbi algorithm sucessfully provides a passable bass harmonization for a full bass line. We notice that the algorithm provides a particularly conservative, and not very diversified bass line, which stresses the tonic, subdominant and dominant functions over all other chords. While the resulting harmonization is technically decent, it is musically not the most interesting thing to do.

3.3 Performance of the expert system in voice leading

For the most part the voice leading algorithm we constructed does a passable job. We did not implement any constraints against parallel fifths and octaves, so these are occasionally present in the output.

3.4 Neural network output

The neural network takes on various behaviors throughout its training. While it quickly seems to have working knowledge of how chords are assembled (remember that this network is manipulating individual notes, so is tasked with figuring out for itself how to construct harmony) it takes time for it to figure out an appropriate register to work in. Notably, voice leading from one chord to another is usually smooth - albeit with many parallel fifths and octaves - and particularly the bass lines stand out as being strong. We observe senses of larger tonal structure around cadential figures, though the effect is diminished by the fact that the neural network rarely writes itself a conclusion.
After the training of the network, actually generating output is extremely fast. With some modifications, it would also be possible to configure this network so that it responds in real time to a bass given by the user. From the observed data, it is bound to take more challenging approaches to harmonizing than the HMM and particularly the Viterbi solution to the bass line.

4. DISCUSSION

4.1 Flaws in HMMs

HMMs are a strong abstraction. Once trained, they are easy to understand and play around with. However, there are inherent issues with the abstraction that make it difficult to treat as a fully-capable musical decision. Firstly, the very core of the Markov property forms a limit on how well it can deal with longer dependencies in the music. Though the n-grams mitigate this issue, they form a problem of their own as they begin to quote the source data. To avoid overfitting to the data, we can employ smoothing methods to give weight to progressions not seen in the data. However, this in turn diminishes the importance of actual observed data. While for a single model this is not dramatic, it would be problematic when using Markov distributions to compare the harmonic patterns of Bach to other composers. [9]
There are some academic efforts to augment the HMM knowledge through external constraints, or by training it on multiple musical viewpoints simultaneously. [10] We would have to look into these methods and see whether implementing them in our case would also lead to advances in musical tension.
Regarding the Viterbi solution to the bass line, we notice that the actual nature of the algorithm makes for unmusical decisions. Musically, one would hope for a certain diversi-

fication or "surprise" to create novelty, but the algorithm's approach to likelihood makes these surprises no-go areas. We searched for ways to locally relax the algorithm but did not find a way to do so in salient ways. Overall, though, it may not be an ideal approach to what we hope to achieve with an unfigured bass harmonizer. We envisioned much of these techniques to be helpful to students learning along with the system how to perform extemporized harmonizations, and the Viterbi algorithm in itself does not lend itself well to user input.

4.2 RNNs effective yet obtuse

We could not in the time provided adequately collect a large enough dataset and leave it to train for long enough to explore the full capability of the neural network approach. However, given that we gave it far less data than recommended, the results were still quite promising. Nevertheless, given that there is no real way to interrogate the system's internal representation beyond what it is outputting between iterations. Karpathy describes in his blog a method to visualize how each neuron is firing, but this is arduous to set up and does not always give straightforward correspondences. Furthermore, while there are good ways to manipulate a HMM (by manually setting transition probabilities) there is not an evident way to adjust a neural network in training. With the lack of manipulability and the relative lack of data (compared to the usual datasets these programs work with) it may be tricky to expand the neural network approach to anything beyond compositional styles for which large datasets already exist. Even then, it remains to be seen whether it can adhere to a particular style successfully.

4.3 Applications

The more prosaic performance of the HMMs and its adherence to data make it a more likely candidate to illustrate interactively how the underlying harmony works - we can visualize Bach's harmonic language as a directed graph. The interactivity and the way that it relies on historical precedent make it an interesting tool to learn music theory interactively. Meanwhile, the neural network is primarily interesting for its autonomy but is not ideal in manipulating creatively. More technical tools need to be developed before it can be employed in a more open-ended artistic practice.

5. CONCLUSION

We explored two different configurations to use aspects of machine learning to generate chordal progressions and harmonize given bass lines. Overall, the results were decent from a theoretical standpoint yet not always satisfying musically. Hidden Markov Models were satisfying in their relative simplicity and close link to the original training data yet are unable to adequately deal with generating longer-term dependencies. Most algorithms dealing with HMMs also focus on optimal solutions, which may not be ideal in a musical context. However, as a pedagogical/statistical model of a corpus there may be some interest in exploring them further.

Our expert system in voice leading performed mechanical operations, and does not provide output that is very musically informed. Despite the presence of well-established rules regarding voice leading, it might not be the best overall method to generate these progressions.

RNNs seem more promising for music geneation: even with a small dataset and a simplistic representation the network began to show promise of performed understanding of the principles of tonal harmony. Though it has not fully demonstrated this yet, the RNN strategy does incline towards incorporating longer dependencies more easily. However, its weaknesses are its internal obscurity in its decision process as well as the large volume of data it requires before it would be performant. Where the HMM could serve as a pedagogical/analytical tool, this neural network acts more as a standalone generative system. It would be helpful to have more control or insight into its internal workings before it could be fully incorporated into a compositional pipeline.

Overall, though the methods examined here are all exciting in their own way, there are still vast gaps between human and computational compositional output. We do not see this gap as a failing on computational part, but rather view this discrepancy as an exciting area to investigate where computers and composers could possibly meet.

6. CODE

The code we wrote for this project can be found at `https://github.com/adrbenn/m21`.

7. REFERENCES

[1] D. Quick, "Generating music using concepts from schenkerian analysis and chord spaces," Yale University, Department of Computer Science, Tech. Rep. YALEU/DCS/RR-1440, May 2010.

[2] E. Özcan and T. Erçal, *A Genetic Algorithm for Generating Improvised Music*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 266–277. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-79305-2_23

[3] R. N. Lichtenwalter, K. Lichtenwalter, and N. V. Chawla, "Applying learning algorithms to music generation."

[4] S. Dubnov, G. Assayag, O. Lartillot, and G. Bejerano, "Using machine-learning methods for musical style modeling," *Computer*, vol. 36, no. 10, pp. 73–80, Oct. 2003. [Online]. Available: http://dx.doi.org/10.1109/MC.2003.1236474

[5] G. Hadjeres and F. Pachet, "Deepbach: a steerable model for bach chorales generation," *CoRR*, vol. abs/1612.01010, 2016. [Online]. Available: http://arxiv.org/abs/1612.01010

[6] music21.corpus. [Online]. Available: http://web.mit.edu/music21/doc/moduleReference/moduleCorpus.html

[7] M. Gogins, "Dmitri tymoczko: A geometry of music: Harmony and counterpoint in the extended common practice." *Computer Music Journal*, vol. 36, no. 1, pp. 83–85, 2012. [Online]. Available: http://dblp.uni-trier.de/db/journals/comj/comj36.html#Gogins12

[8] Andrej, "The unreasonable effectiveness of recurrent neural networks." [Online]. Available: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

[9] M. G. Bergomi, A. Baratè, and B. Di Fabio, "Towards a topological fingerprint of music," in *Proceedings of the 6th International Workshop on Computational Topology in Image Context - Volume 9667*, ser. CTIC 2016. New York, NY, USA: Springer-Verlag New York, Inc., 2016, pp. 88–100. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-39441-1_9

[10] R. P. Whorley and D. Conklin, "Music generation from statistical models of harmony," *Journal of New Music Research*, vol. 45, no. 2, pp. 160–183, 2016. [Online]. Available: http://dx.doi.org/10.1080/09298215.2016.1173708