# DeepFugue: a model to generate Baroque-style fugues

Aditya Chander (ac888@stanford.edu), Marina Cottrell (mborsodi@stanford.edu), Samantha Silverstein (selinons@stanford.edu)

## Introduction

The fugue is a musical form that is considered to exemplify a composer's prowess in balancing harmony, counterpoint, and long-term musical structure. It is an imitative genre of classical music, whereby a musical theme, known as the "subject", is the basis for the entire composition. The subject is passed from part to part, and is heard alongside "countersubjects", other melodies that harmonise appropriately with the subject, whilst demonstrating musical independence and clarity. From the Baroque era onwards (c. 1600 to the present day), fugues have been used as a pedagogical tool for composers and performers alike. We seek to continue this tradition with newly-generated fugues that can be used for similar purposes, as well as creating a new corpus of works for performance.

Our aim is to produce fugues in the style of famed baroque composers, such as Bach and Händel. The input to the model is the desired fugue subject, and the output is a fugue utilising this subject as its principal thematic content. The fugue data comes primarily from the Bach, Handel and Pachelbel compilations from www.kunstderfuge.com, in MIDI format.[1] Our output, also in MIDI format, can be played using software such as Sibelius, MuseScore and GarageBand. We are using a recurrent neural network (RNN) model, with LSTM units to store information about previous notes. This is critical for music generation, in order for the melodies and harmonies to be generated with a sense of context and continuity. The primary source of inspiration for our model is the DeepBach model, used to generate chorales, which are four-part harmonisations of a particular melody.[2]

## Dataset

As mentioned above, the data is in the form of MIDI files. MIDI files are a simple way of structuring musical information, and principally contain:

- the channel number (from 0 to 15),
- the MIDI pitch number (from 0 to 127),
- the status of the note (whether the note is starting or finishing), and
- the velocity (loudness) of the note (from 0 to 127).

This information is embedded into a track chunk, containing the time value of the musical event.[3]

The fugues themselves are primarily written by Johann Sebastian Bach. They are considered by many eminent musicians and musicologists to epitomise fugue composition in terms of their complexity and expressivity. We have taken examples from:

- the *Well-Tempered Clavier* (volumes 1 and 2), a set of 48 preludes and fugues in all the major and minor keys,
- the organ preludes and fugues, and
- various free-standing works for keyboard instruments.

The data also includes similar works by other Baroque composers, such as Georg Frideric Händel and Johann Pachelbel, as well as later composers writing fugues in a similar style, such as Felix Mendelssohn and Robert Schumann. The MIDI files are found on www.kunstderfuge.com.

Of these works, we have kept only the fugues that are composed in four voices. This is to ensure stylistic unity in the dataset. Since we are building on a model that was originally trained on chorales, this ensures that we can maintain a similar architecture in our own model.

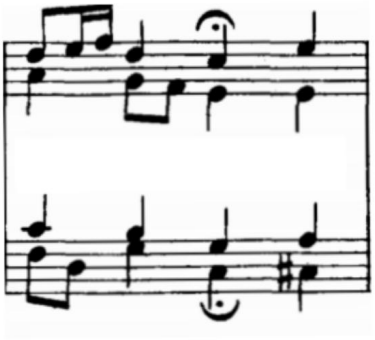
---

[1] "THE LARGEST CLASSICAL MIDI COLLECTION," Kunstderfuge | Free Classical Midi Files, accessed May 21, 2018, http://www.kunstderfuge.com/.

[2] Gaëtan Hadjeres, François Pachet, and Frank Nielsen, "DeepBach: a Steerable Model for Bach Chorales Generation", (paper presented at Proceedings of the 34th International Conference on Machine Learning, Sydney, 2017).

[3] Jean-Pierre Briot, Gaëtan Hadjeres, and François Pachet, *Deep Learning Techniques for Music Generation – A Survey* (Paris: Sorbonne Universités, 2017), p. 18-19.

We have parsed the MIDI data in a similar fashion to how the chorale data in DeepBach was parsed. Figure 1 shows an example representation of music from a chorale in DeepBach.



(a)

```
D5,__,E5,F5,D5,__,__,__,C5,__,__,__,E5
A4,__,__,__,G4,__,F4,__,E4,__,__,__,E4
C4,__,__,__,B3,__,__,__,G3,__,__,__,A3
F3,__,D3,__,G3,__,__,__,C2,__,__,__,C#2
 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, 1
 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0
```

(b)

*Figure 1* (a) Extract from a chorale by Johann Sebastian Bach. (b) DeepBach encoding of this musical information.

The encoding above features six distinct channels, which are:

- the soprano (highest) voice,
- the alto (second highest) voice,
- the tenor (second lowest) voice,
- the bass (lowest) voice,
- the sixteenth-note subdivision of the quarter-note pulse, and
- whether the note has a fermata (pause) on it.

We use the same encoding for our data, without the fermata channel, since fermatas in fugues do not, in general, reveal salient information about the harmony or counterpoint. We used the '_' token to represent a sustained note, and a 'R' for a rest.

In order to encode the fugue subject, we found the music in each fugue up to the point that the second voice enters, and isolated this music in a single encoding. This is an approximation for a fugue subject – occasionally, fugues feature a short additional bridge passage for a smooth transition into the second entry, which is not truly part of the subject. Nonetheless, these bridge passages are short, and could be incorporated into a fugue subject input if so desired.

We padded each fugue with a 'fin' (end-of) token up to the end of the longest fugue, and did the same for fugue subjects. This was to ensure the dimensions were consistent for each input. We also augmented the dataset by transposing the fugues and their subjects into the twelve different possible tonal centres, or 'keys', of the Western tonal system. This was achieved easily, by subtracting up to 5 and adding up to 6 from each MIDI pitch value.

Overall, we ended up with 612 fugues in the dataset. Due to the highly idiosyncratic nature of the output, we felt that testing wasn't suitable for the task, and we preferred to judge our results more qualitatively ourselves. Testing only told us how close the generated fugues were to other fugues, not how fugue-like they actually were

## Architecture

We initially considered models inspired by the DeepBach architecture, and the BachBot architecture.[4] Both of these models sought to generate chorales, or four-part harmonisations of a melody. The DeepBach architecture can be seen in figure 2.

[4] Feynman Liang, *BachBot: Automatic Composition in the Style of Bach Chorales*, Master's thesis, University of Cambridge, 2016 (Cambridge: University of Cambridge, 2016).

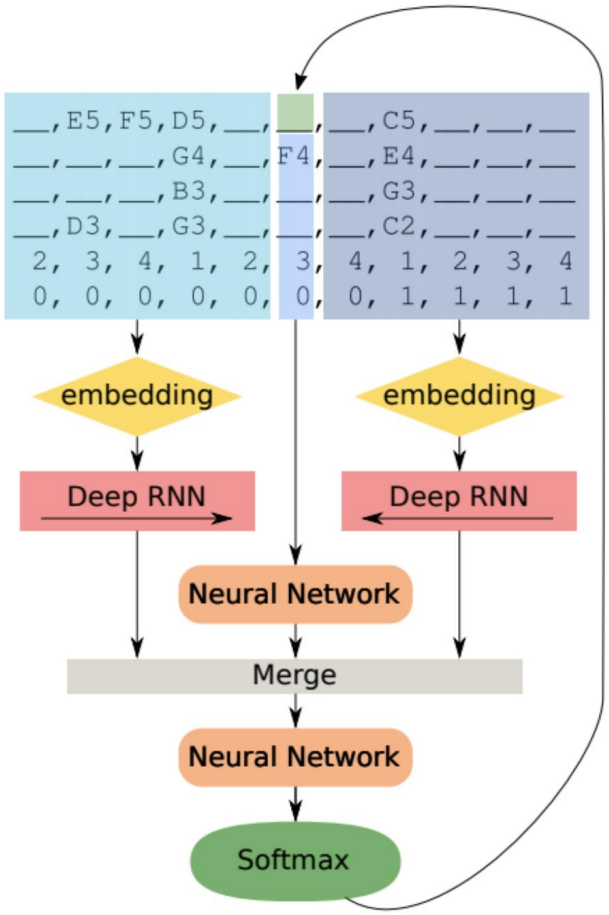



*Figure 2* Graphical representation of DeepBach's neural network architecture for the soprano prediction at the current timestep.

We found this architecture to be quite convoluted, and difficult to scale from chorales to fugues. The main issues were as follows:

- The chorale outputs have the same number of timesteps as the inputs, while this was not the case for fugues. We tried padding the fugue subjects up to the length of the longest fugue in our dataset, but found that this disguised too much information, since fugue subjects are significantly shorter than fugues.
- Each of the four neural networks in the architecture had to be retrained with the new data. Computationally speaking, this was very costly.

As such, we had very low performance, and most of the outputs consisted entirely of sustain ('_') tokens, or rests ('R').

Hence, we designed a new architecture, based on the Neural Machine Translation with Attention model from the NLP/Sequence Models course on Coursera.[5] We used the Keras framework.[6] Figure 3 shows our chosen architecture, including the specifics of the attention mechanism.

In our model, these variables represent the following:

- Tx is the number of sixteenth-note beats in the longest fugue subject.
- Ty is the number of sixteenth-note beats in the longest fugue multiplied by 4 (the number of voices in the fugue).
- The one-hot vectors $x^{<t>}$ denote the MIDI pitch value at the current time step. This vector can also represent a rest, a sustained note, or the end-of-subject 'fin' token.
- The output vectors $\hat{y}^{<t>}$ are softmax vectors, predicting the probability of a particular note in a particular voice at each time step.

[5] Andrew Ng, "Neural Machine Translation with Attention", Coursera – NLP/Sequence Models, accessed May 25, 2018, https://bit.ly/2M40By2.
[6] Keras, accessed May 25, 2018, https://keras.io/.

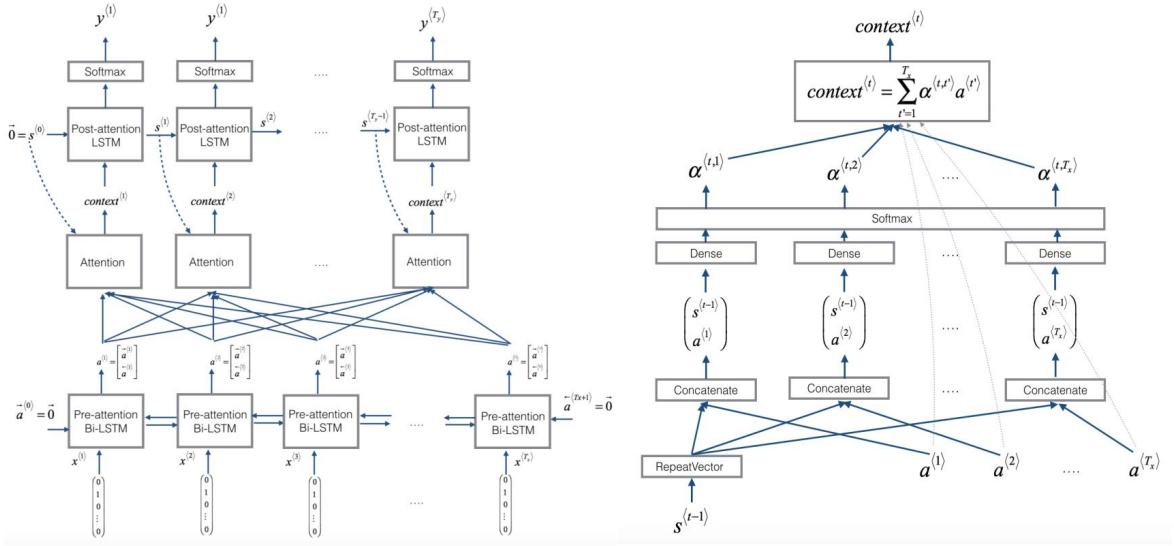



*Figure 3* RNN architecture for our model, and the attention mechanism in more detail.

We used a binary cross-entropy loss function:

$$\mathcal{L}(\hat{y}, y) = -\sum_{i=1}^{T_y} y_i \log(\hat{y}) + (1 - y_i) \log(1 - \hat{y}_i)$$

## Results

We recorded the results after 10, 100 and 1000 epochs of training, using the melody from "Twinkle Twinkle Little Star" as our input. After 10 epochs, we found that our model was essentially outputting random notes, with very little imitation in general, and little regard for the general register in which each voice of the fugue operates – for instance, the bass and tenor voices were outputting very high notes which would almost never appear in this voice.

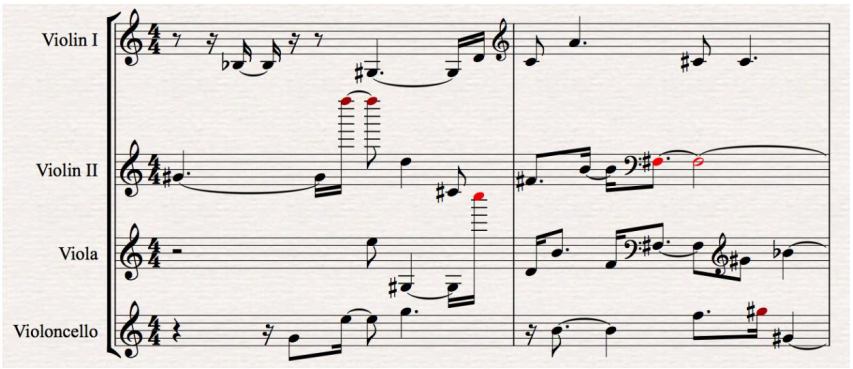



*Figure 4* Sample output after 10 epochs.

After 100 epochs, the imitative character was more apparent, and the outputs took on a more appropriate rhythmic profile. Harmonically, however, the outputs left a lot to be desired – they sounded quite atonal and not in-keeping with baroque harmony.

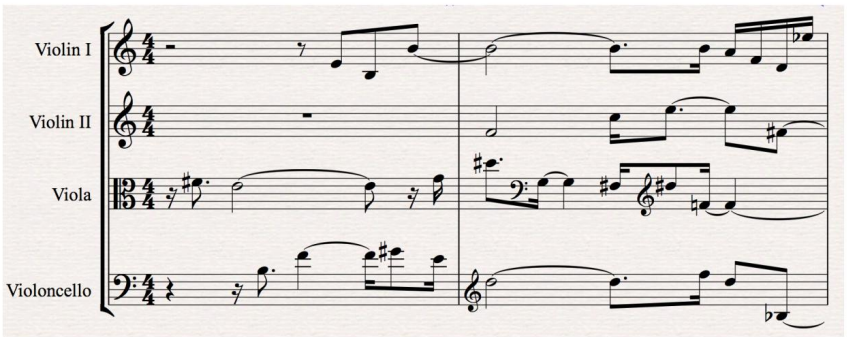



*Figure 5* Sample output after 100 epochs.

After 1000 epochs, there was significant improvement. The harmony was quite tonal, and the voicing was much more in-keeping with what one would expect from each voice. The imitative character and rhythmic profile became stronger, though there was no real allusion to our initial input of the "Twinkle Twinkle" melody, which one would expect to be preserved in a fugue.

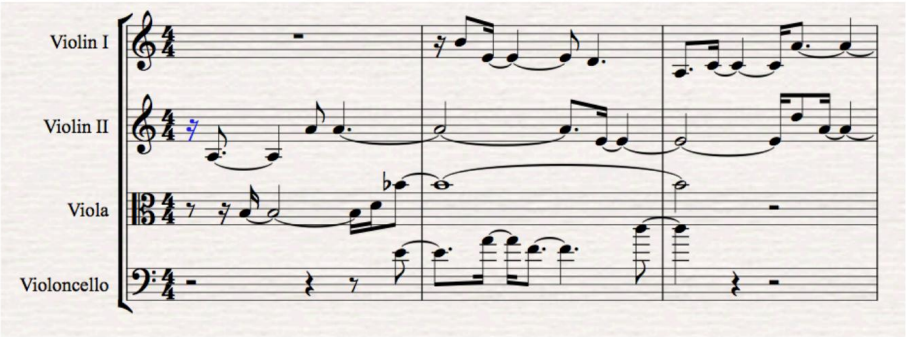



*Figure 6* Sample output after 1000 epochs.

## Conclusions and future work

The main issues we had were with the sparsity of the data, as well as the complexity. While we augmented our dataset, the lack of fugue data specifically made it challenging to build a model that did not overfit what we had. Since Aditya is skilled in baroque-style fugue composition, he added a few extra examples of his own to the dataset, and this helped somewhat. With more time, he would ideally have written some more – a good quality, four-voice fugue takes him on average four hours to compose, and an hour to polish.

We believe that adding some harmonic information to the input as a feature might improve our model. For a client, adding the harmony to a standalone fugue subject is not necessarily intuitive, so this could be achieved by first passing the melody through a model akin to DeepBach or BachBot, followed by inputting this output into the fugue generator.

Formatting our data correctly for the architecture was a challenge. Many of the fugues from www.kunstderfuge.com were formatted inconsistently, in that individual voices in a fugue were sometimes compressed into a single channel. We were forced to throw out many of these examples, as there was very little we could do to separate the voices in the way that the composer intended for them to be heard. However, once we had our data in the appropriate format, it was a small step to convert them into one-hot vectors for training.

We hope that our work will be relatively easily extensible to other genres of baroque music that are related to four-part fugues. These include:

- Fugues in different numbers of voices
- Fugues with more than one subject (so-called 'double fugues')
- Other contrapuntal forms, such as canons (exact imitation between the parts)

We also hope that our work can contribute to musicological research aimed at reconstructing incomplete fugues, or works containing fugal sections. This is applicable, for example, to Johann Sebastian Bach's monumental work *The Art of Fugue*, which was left incomplete at the time of his death.

## Contributions

Marina's main role was to write algorithms to parse the data and decode the outputs into a meaningful form. Sam and Aditya consulted on this and wrote code to reshape the data appropriately at each stage of the process, to fit the model architecture.

Sam and Aditya focused on analysing the fugues and building the model architecture, with Marina consulting and training on her GPU. Aditya wrote some fugues for the dataset, as well as the write-up and the poster.

## References

"THE LARGEST CLASSICAL MIDI COLLECTION," Kunstderfuge | Free Classical Midi Files. Accessed May 21, 2018, http://www.kunstderfuge.com/.

Briot, Jean-Pierre; Hadjeres, Gaëtan; and Pachet, Françoıs. *Deep Learning Techniques for Music Generation – A Survey.* Paris: Sorbonne Universités, 2017, pp. 18-19.

Hadjeres, Gaëtan; Pachet, Françoıs; and Nielsen, Frank. "DeepBach: a Steerable Model for Bach Chorales Generation". Paper presented at Proceedings of the 34th International Conference on Machine Learning, Sydney, 2017.

Keras. Accessed May 25, 2018, https://keras.io/.

Liang, Feynman. *BachBot: Automatic Composition in the Style of Bach Chorales.* Master's thesis, University of Cambridge, 2016 (Cambridge: University of Cambridge, 2016).

Ng, Andrew. "Neural Machine Translation with Attention", Coursera – NLP/Sequence Models. Accessed May 25, 2018, https://bit.ly/2M40By2.