BachNet

Johan von Hacht **David Johansson** **David Stevens**

Abstract

The task involved training a model that is capable of generating harmonies given a melody. The final model employed a sequence-to-sequence structure where each harmony gets generated by separate GRU encoders and decoders before being combined with a fully connected output layer. This model managed to generate harmonies that closely resemble human-created arrangements regardless of the key in which the melody is played, largely due to the employed data augmentation strategy. However, even though Bach's chorales were used as training data, the generated compositions do not, necessarily, sound reminiscent of the composer's work as they employ more modern syncopated rhythms. This is due to the one-hot encoding that was used, showing that special care should be taken when designing the numerical representation of music.

1 Introduction

Composing well-formed music requires the understanding of musical rules as well as creativity, characteristics that are not necessarily associated with computers. Therefore, the act of composing music has intrinsically been regarded as solely a human activity for very long. However, due to advancements within the field of deep learning, the lines between what a computer is and is not capable of in terms of music composition gets blurred. Several studies have been made regarding generative models capable of composing music with promising results.

This report serves the purpose of expanding upon these findings by trying to create a model that generates choral harmonies given a melody. The problem at hand is important considering that a generative model could help facilitate the composition process by creating harmonies for you. This could also help lowering the threshold for music creation and inspire newcomers to experiment in the genre. As stated by Hild et al. [1992], the results could also "shed some light on an aspect of human creativity that doesn't seem to be describable in terms of symbols and rules".

After testing several different approaches, the most successful would end up being a three-part sequence-to-sequence model where the harmonisation of each voice type gets determined by their own encoder and decoder. Each encoder and decoder is made up of a Gated Recurrent Unit (GRU) where both the input and output of the model as a whole use one-hot-encoding. In order to optimise the results of the model, several experiments were performed such as a coarse search that determined the optimal value for the dropout rate. In the end, the harmonies that are generated by BachNet sound musically pleasing and interesting. The results may not closely resemble the compositions of Bach but still shows signs of the kind of structure that a human could write.

2 Related work

The task of generating harmonies given a melody was explored in a study by Hild et al. [1992]. Within its task definition, the melody of a composition is characterised by the soprano voice type while alto, tenor and bass makes up the harmonisation. The study proposes an approach for approximating the function that generates chorale harmonies referred to as HARMONET. The results were found to be on the level of an "improvising organist" and concluded that the use of neural networks outperformed other classification techniques such as decision trees and nearest neighbour classification.

Project report for DD2424

One of the more successful approaches in regard to generating music has been sequence models such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). These types of models has managed to achieve notable results in regard to specific representations of music such as folk melodies as evident by Sturm et al. [2016]. Chung et al. [2014] compares the performance of the LSTM and GRU. The GRU model is similar to that of a LSTM except that it makes use of fewer parameters by not having to keep track of a memory cell. Although the simpler structure, the performance of the GRU was found to be comparable to LSTM in terms of music modelling.

Pascanu et al. [2013] explores the difficulties pertaining to training Recurrent Neural Networks such as LSTMs and GRUs. Their research found that regularisation techniques in the form of L1 and L2 penalties had negative effect on generative models, especially in terms of exhibiting long-term memory.

An approach that differs from the aforementioned RNNs is brought up by Huang et al. [2019a]. In this study, a Convolutional Neural Network (CNN) is trained that serves the purpose of completing partial musical scores. The trained model ended up being capable of yielding impressive results that were difficult to distinguish from human-made compositions.

3 Data

The following section will describe the dataset that the final model was trained on, performed preprocessing, and previous projects where it has been utilised.

3.1 Description

The dataset that the project has been based on is referred to as *JSB Chorales*. The dataset consists of 382 different musical compositions by Johann Sebastian Bach quantised to 16th notes, which are divided into three sets organised as training (229 pieces), validation (76 pieces) and testing (77 pieces). Each time step of the composition is represented as four numbers. The numbers constitute the MIDI pitch values of all four different SATB voice types in the form of *soprano*, *alto*, *tenor* and *bass*, totalling 45 different pitches.

The dataset can be found at `www.github.com/czhuang/JSB-Chorales-dataset` where it is also provided in quarter or 8th note quantisations. These quantisations decide which fraction of the beat that notes can be played on, essentially deciding the rhythmic "resolution" of the music. This means more notes can be played during a 16th note quantisation compared to a 8th note quantisation. In this report, the utilised dataset abided by a 16th note quantisation.

3.2 Preprocessing

In order to make the data easier to work with, it had to be preprocessed. The preprocessing primarily consisted of transforming it into a one-hot encoded format. In the one-hot encoded input data, the pitch value of a specific voice type is represented through a column vector of length 46. The first 45 indexes represents the possible pitches between 36 and 81 while the last index makes up the possible silent pitch. This format will be passed into the model as the soprano voice type.

3.3 State-of-the-art

The JSB Chorales dataset has laid the foundation for different studies with one of the more notable ones being carried out by Huang et al. [2019a]. The findings made by Huang, more specifically the produced model, would subsequently be utilised when creating a Bach-themed Google Doodle that celebrated the 334th birthday of the composer [Huang et al., 2019b].

4 Methods

A few different approaches were tested for solving the problem. First, the final implementation of BachNet will be described and, after that, data augmentations and alternative approaches.

4.1 BachNet

Seq2seq is a machine learning method where sequences are converted from one domain to another. It is often used in machine translating where one sentence is translated into another language [Sutskever et al., 2014]. The model is inspired by this approach since creating a harmony from a melody could be seen as translating between two "musical" languages. Similar to seq2seq, BachNet consists of encoding layers whose final state becomes the initial state of the decoding layers. However, the approaches differ because the BachNet model outputs sequences directly, instead of using the recursive inference loops that are conventional to seq2seq models. Furthermore, the encoder GRUs output both the initial state for the decoders and the generated state sequence, providing a simplified form of access to the original melody than with attention.

The BachNet model consists of eight layers with $10\,339\,374$ trainable parameters, seen in figure 1. The input is first passed through the melody encoder. The output of this encoder is then passed to the three different harmony encoder-decoders: alto, tenor and bass. The intent with this structure is for the model to be able to learn the different characteristics of the harmonies separately at first. However, since they are combined into a final shared dense layer, the model should in theory also learn how the melody and harmonies interact with each other. All of the encoders and decoders are GRU-layers. These layers were used instead of LSTMs due to the findings in Chung et al. [2014] showing that they have similar performance while being simpler.

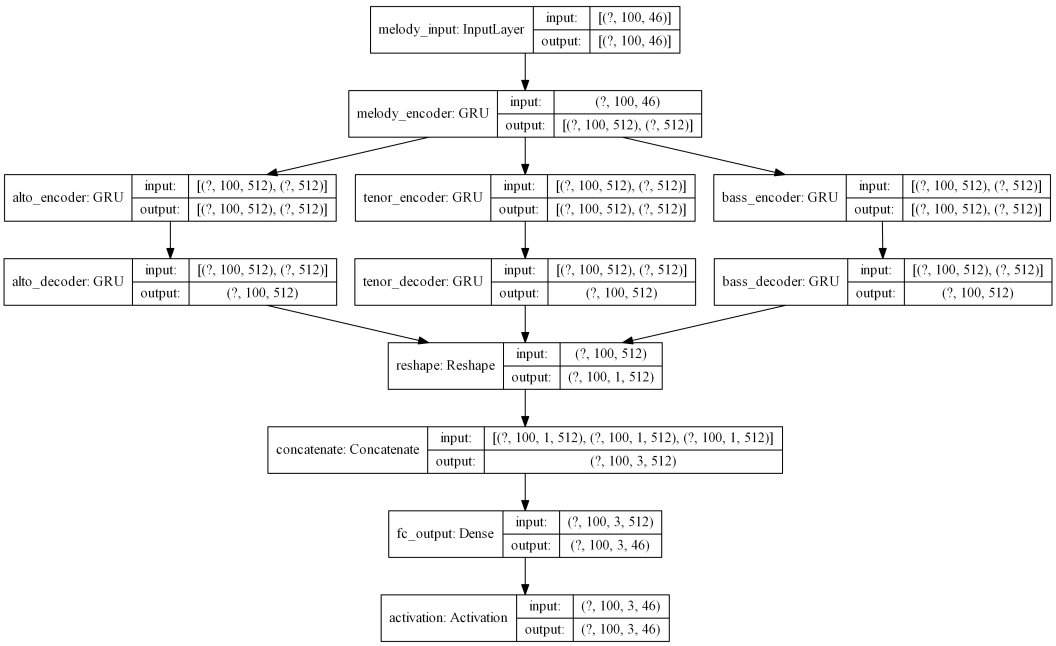


Figure 1: Overview of the BachNet model.

Without any form of regularisation, the network was prone to overfitting. To negate this, dropout was deployed equally to each GRU layer with a factor of 0.3 as decided in section 5.1 and training was stopped when the validation loss stopped improving.

4.2 Data augmentations

Note transposition One difficulty in generating harmonies is that melodies can be performed in different keys while retaining the same harmonic functions. This has been solved by transposing all training data to the same key [Liang et al., 2017], or including the key as metadata [Hadjeres et al., 2017]. We instead took a different approach, in which all training data was transposed to every key that was possible while still remaining within the vocal ranges of the four parts. This lead to a more resilient model that does a better job of harmonising music in any given key while still respecting the vocal ranges of the performers. The total dataset increased in size from 382 to 2888 pieces through this augmentation.

Cropping data Our model was designed to work with a fixed input length that was set to the minimum length of a song in the training dataset, being 100 sixteenth notes. In order to avoid losing training samples from cropping all other pieces to this length, we generated every non-overlapping subsequences of 100 sixteenth notes from the training data and used this as our dataset. This increased the size of the training dataset from 1692 samples to 3113 samples.

4.3 Alternative approaches

Single LSTM The initial model implementation for harmony generation utilised a LSTM. A LSTM was used instead of a RNN in order to avoid the vanishing gradient problem. Promising results could be made regarding the generation of one single harmony such as bass by employing only one LSTM as it was capable of overfitting. However, the compositions would not sound coherent when combining the different generated harmonies which prompted the use of a different model that generates all harmonies at once. When using a single LSTM to generate all three harmonies, the results pointed to the model underfitting.

Encoder-decoder LSTMs A sequence-to-sequence model was thought to be a good option when generating all harmonies at once considering that the end-result is based on an intermediate state representing the whole input data. As each harmony is generated from this state, it was believed that they would sound more coherent together. However, the model seemed to be underfitting. Additionally, problems occurred with note selection from the soft-max model output. With argmax note selection, the composition contained mostly repeated notes which was not desirable. Another strategy for note selection was to choose randomly based on the soft-max distribution but this led to noisy melodies.

Encoder-decoder with Attention With the purpose of making the encode-decoder structure more effective, the inclusion of Attention was explored. With an Attention-mechanism, more specifically Bahdanau Attention [Bahdanau et al., 2014], the training is able to pay more attention to specific parts of the input. Furthermore, the model utilised Teacher Forcing as a means of converging faster. This model ended up underfitting which points to it not being capable of capturing the underlying structure of the data at hand. This might also have been due to an implementation error from the increased complexity of the model.

Text-encoded seq2seq In order to translate our task into the conventional seq2seq domain of text strings, we experimented with using the same textual encoding of the music as Liang et al. [2017]. A simple seq2seq model without attention was then tested on both a character-level and word-level. The character-level model did not grasp the syntax, leading to invalid output files, while the word-level model did not produce output files that matched the larger scale structure, such as having three notes per time step.

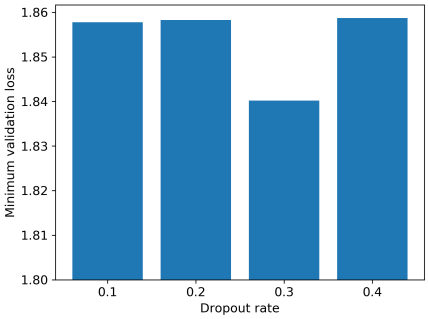
Repeat token The first iterations of the BachNet models did notreplicate held notes well which made the harmonies sound noisy. The most likely cause of this was because a held note was represented as a sequence of identical repeating notes. A proposed solution to this problem was to implement hold tokens in the input data. A hold token meant that the previous note should be repeated. However, since the input data now consisted of a majority of hold tokens, there was less focus on the actual notes. As a result, the model learnt to output hold tokens while disregarding the actual notes. Subsequently, this encoding was abandoned in favour of increasing the model complexity instead of changing the encoding.

5 Experiments

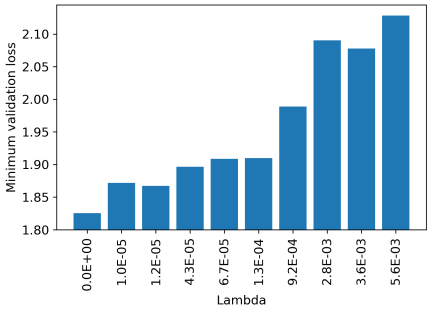
The following section will outline different experiments that were performed in order to improve the model performance. The performance was primarily assessed by interpreting the effect different changes had on the validation loss.

5.1 Hyperparameter search

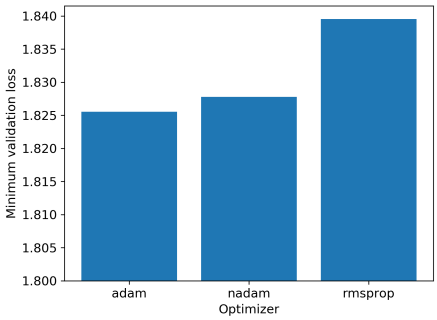
To improve the performance of the model, a hyperparameter search was performed. The experiments for each parameter was done independently of each other and the performance was measured by the validation loss achieved by the model.



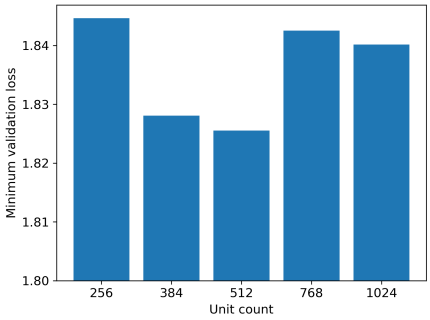
(a) Validation loss with tested dropout rates.



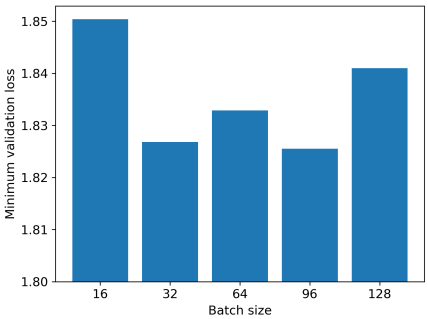
(b) Validation loss with tested lambda values.



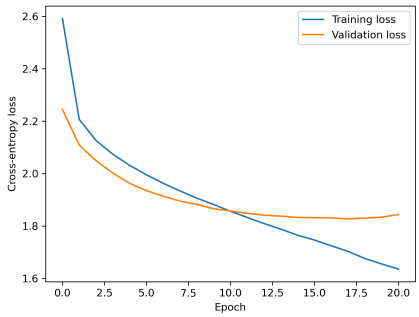
(c) Validation loss with different optimisers.



(d) Validation loss with different unit counts in the GRUs.



(e) Validation loss with different batch sizes.



(f) The final training history

Figure 2: Results of the hyperparameter search

Dropout Tested values for the amount of dropout were 0.1, 0.2, 0.3 and 0.4. As seen in figure 2a, the best dropout value for the model was 0.3.

L2-regularisation We experimented with L2-regularisation applied to all GRU kernels and performed a course-search to find a suitable value for lambda. During the search, 10 different lambdas were tested by sampling from 10^k with random values of k, where $k \in \mathcal{U}(-5, -2)$. The results can be found in figure 2b. From these results, it became apparent that no suitable value could be found considering that the validation loss was increased for all the lambda values. This pointed to

L2-regularisation not being effective in terms minimising overfitting for our model and was, therefore, left out. These results stays in line with the aforementioned findings by Pascanu et al. [2013].

Optimisers Three optimisers were compared by training the model until convergence: Adam, Nadam and RMSProp. The best performing optimiser was Adam which can be seen in figure 2c. SGD, Adadelta, Adagrad, Adamax and Ftrl were also tested, but took too long to converge on a result and were aborted early.

Latent units Figure 2d points to 512 being the best unit count.

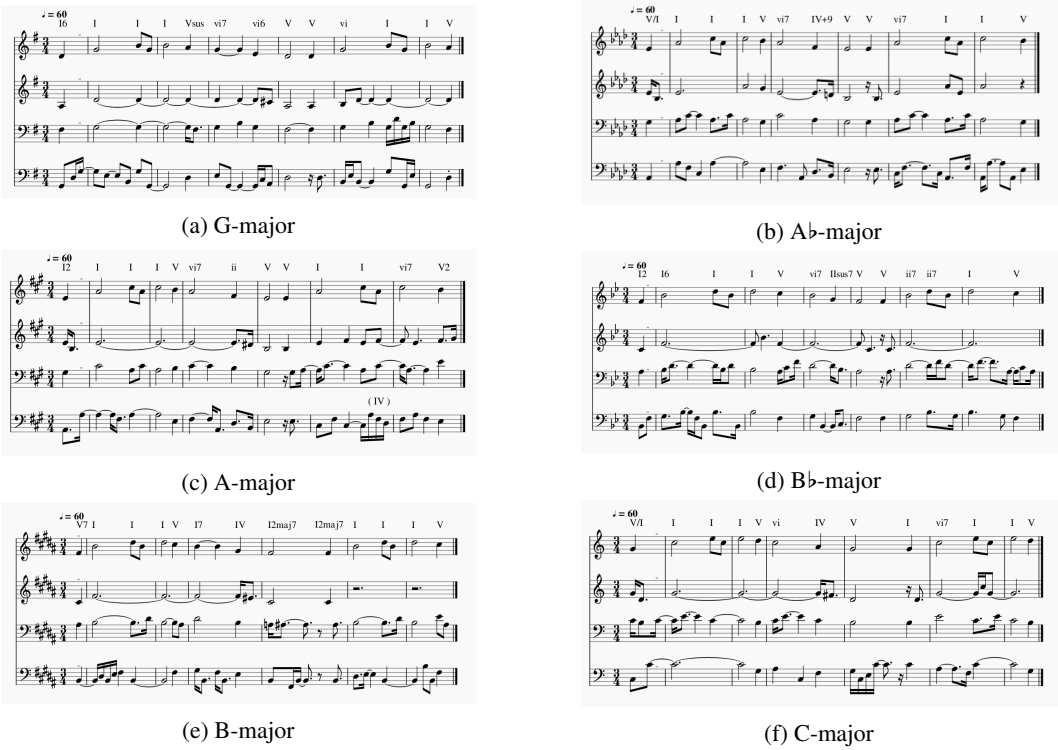
Batch size Figure 2e shows that batch size of 96 performed the best on the validation data.

5.2 Final training

With all of the above hyperparameters set, we achieved a lowest validation loss of 1.8267 as shown in figure 2f when training on both the training and test data and stopping training when the validation loss stopped improving.

5.3 Compositions

Transpositions One of the major goals of performing transpositions of the music as data augmentation was to make the model better at identifying the key of a melody and resilient to the same melody being played in different keys. In order to test this, we harmonised *Amazing Grace* in six semitone separated keys and analysed the harmonic function of the chosen chords. This analysis was done with Roman numeral analysis, a system for abstracting chord progressions to make them independent of the key. If the model harmonises the melody with the exact same chords, the same Roman numerals would be presented for each key. The full arrangements are presented in figure 3, while the Roman numeral analysis is presented in table 1.



(a) G-major

(b) A♭-major

(c) A-major

(d) B♭-major

(e) B-major

(f) C-major

Figure 3: Six harmonisations of *Amazing Grace* in different keys

All of the arrangements remained in key with the exception of rare accidentals used in transitions between chords. The Roman numeral analysis in table 1 shows that while there was some difference

Table 1: Roman numeral analysis of the arrangements in figure 3

Chord number	Chosen chord in key					
	G	A♭	A	B♭	B	C
1	I6	V/I	I2	I2	V7	V/I
2	I	I	I	I6	I	I
3	I	I	I	I	I	I
4	I	I	I	I	I	I
5	Vsus	V	V	V	V	V
6	vi7	vi7	vi7	vi7	I7	vi
7	vi6	IV+9	ii	IIsus7	IV	IV
8	V	V	V	V	I2maj7	V
9	V	V	V	V	I2maj7	I
10	vi	vi7	I	ii7	I	vi7
11	I	I	I	ii7	I	I
12	I	I	vi7	I	I	I
13	V	V	V2	V	V	V

in ornamental tones, the melody was harmonised in mostly the same way regardless of which key it was in. This indicates that the model has learned to analyse the core harmonic function of the melody rather than the specific notes.

Furthermore, the harmonies that were written remain within the vocal range of each of the four parts in all six keys. This indicates that the model has also learned the vocal range of each part and changes its harmonisation to respect these limits, meaning that generated harmonies in all likelihood could actually be performed by a SATB quartet. That the model could learn this is due to the care that was taken during the transposition phase of the data augmentation, where any transpositions that lead to tones which were outside of the vocal range of the given part were discarded.

Finally, it is interesting to see certain recurring motifs in the arrangements, such as the syncopations in the accompanying parts and the accidental leading into the fourth measure that was present in either the tenor or alto part in all arrangements except one. This is further evidence that the model views the arrangement independently of the key.

5.4 Common flaws

While the model was successful at producing choral arrangements with valid harmonies regardless of key, it did not produce arrangements that were rhythmically similar to Bach when harmonising unseen data. This was largely due to the fact that the model would make frequent use of syncopations, which are essentially nonexistant in Bach's music. This becomes especially noticeable when the model is asked to arrange a melody that are supposed to be tranquil, such as the Swedish hymn *Bred dina vida vingar* which is displayed in figure 4. We believe that the reason why syncopations were so heavily employed is the encoding that was used. Since there was no internal differentiation between notes that were placed on the beat and syncopated off the beat, the total loss would be minimised if all erroneous tones that were placed on a beat are immediately adjusted to the correct tones in the next sixteenth note. This could potentially be avoided by having an encoding where a tone is produced along with the duration of the tone, which would be more similar to the MIDI standard.

6 Conclusions

In this work, we showed that a sequence-to-sequence structure can be used for harmony generation given a melody. The final model ended up being capable of generating some pleasant compositions, albeit quite unlike the work of Bach.

Furthermore, it managed to generate harmonies despite the input melodies being played in different keys while respecting the vocal ranges of the performers. This points to the model having learnt to analyse the core harmonic function, rather than looking at specific notes. The cause of this generality is in all likelihood the employed data augmentation strategy, wherein the data set was transposed to

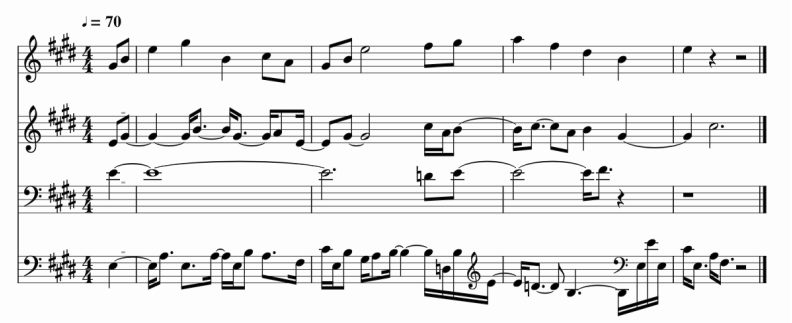


Figure 4: A particularly syncopated arrangement of the Swedish hymn *Bred dina vida vingar*

every possible key that remained within the vocal ranges of the performers. Performing a similar augmentation is likely to prove beneficial to future work in the field.

Future extensions of this work could be to decrease the amount of syncopations caused by the music encoding. An encoding scheme that is more similar to the MIDI standard, where notes are specified along with their duration, could be a good first step. This would also add support for other quantisations, such as triplets. It would also be interesting to experiment with decoupling the different harmonic components, as a trained tenor encoder/decoder component could potentially be reapplied in a different ensemble setting such as a barbershop quartet with minor retraining.

References

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

G. Hadjeres, F. Pachet, and F. Nielsen. Deepbach: a steerable model for bach chorales generation. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1362–1371. JMLR. org, 2017.

H. Hild, J. Feulner, and W. Menzel. Harmonet: A neural net for harmonizing chorales in the style of js bach. In *Advances in neural information processing systems*, pages 267–274, 1992.

C.-Z. A. Huang, T. Cooijmans, A. Roberts, A. Courville, and D. Eck. Counterpoint by convolution. *arXiv preprint arXiv:1903.07227*, 2019a.

C.-Z. A. Huang, C. Hawthorne, A. Roberts, M. Dinculescu, J. Wexler, L. Hong, and J. Howcroft. The bach doodle: Approachable music composition with machine learning at scale. *arXiv preprint arXiv:1907.06637*, 2019b.

F. T. Liang, M. Gotham, M. Johnson, and J. Shotton. Automatic stylistic composition of bach chorales with deep lstm. In *ISMIR*, pages 449–456, 2017.

R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318, 2013.

B. L. Sturm, J. F. Santos, O. Ben-Tal, and I. Korshunova. Music transcription modelling and composition using deep learning. *arXiv preprint arXiv:1604.08723*, 2016.

I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.