AUTOMATIC STYLISTIC COMPOSITION OF BACH CHORALES WITH DEEP LSTM

Feynman Liang
Department of Engineering
University of Cambridge
fl350@cam.ac.uk

Mark Gotham
Faculty of Music
University of Cambridge
mrhg2@cam.ac.uk

Matthew Johnson
Microsoft

Jamie Shotton
Microsoft

ABSTRACT

This paper presents "BachBot": an end-to-end automatic composition system for composing and completing music in the style of Bach's chorales using a deep long short-term memory (LSTM) generative model. We propose a new sequential encoding scheme for polyphonic music and a model for both composition and harmonization which can be efficiently sampled without expensive Markov Chain Monte Carlo (MCMC). Analysis of the trained model provides evidence of neurons specializing without prior knowledge or explicit supervision to detect common music-theoretic concepts such as tonics, chords, and cadences. To assess BachBot's success, we conducted one of the largest musical discrimination tests on 2336 participants. Among the results, the proportion of responses correctly differentiating BachBot from Bach was only 1% better than random guessing.

1. INTRODUCTION

Recent advances have enabled computational modeling to provide novel insights into a range of musical phenomena. One use case is *automatic stylistic composition*: the algorithmic generation of music in a style similar to a particular composer or repertoire. This study explores that goal, restricting its attention to *generative probabilistic sequence models* which are *learned from data*. This model is desirable because it can be applied to a variety of tasks, including: harmonizing a melody (by conditioning the model on the melody) and automatic composition (by sampling a sequence from the model).

The aim is to build a system capable of generating music in the style of Bach chorales such that *an average listener cannot distinguish it from original Bach*. While the method we develop is capable of modeling any multi-part music, we limit the scope of this work to Bach's chorales because: they provide a relatively large corpus, by a single composer, are well understood by music theorists, and are routinely used in the teaching of music theory.

1.1 Related Work

Two well-known difficulties in automatic composition are 1) learning the long-term dependencies required for plausible phrasing structure and motif distribution [31], and 2) evaluating the model's performance rigorously [34]. Addressing the first difficulty, more recent work has reported improvements in learning long-term dependencies by using LSTM [14, 13, 18]. Eck and Schmidhuber [14] used LSTM to model blues music and found that LSTM can indeed learn long-term aspects of musical structure such as repeated motifs without explicit modelling.

Evaluating model performance has proven to be more problematic. In recent work, researchers have begun conducting larger-scale human evaluations. Quick [35] evaluated her rule-based system's outputs on 237 human participants from Amazon's MTurk. Perhaps most relevant to the present study is Collins et al. [6]: a Markov chain expert system for automatic composition. The authors evaluated on 25 participants with a mean of 8.56 years of formal music training and found that only 20% of participants (5 out of 25) performed significantly better than chance. While these prior results are strong, both of these systems relied upon a large amount of expert domain knowledge encoded into the models. In contrast, BachBot leverages minimal prior knowledge and is evaluated on a significantly larger participant pool.

Bach chorales have been a popular corpus for previous work on automatic composition. Early deterministic systems included rule-based symbolic methods [7, 8, 12, 36], grammatical inference [9], and constraint logic programming [39]. Probabilistic models learned from data include the effective Boltzmann machine [3] as well as various connectionist models [37, 38, 24, 31, 15, 27].

Allan and Williams [1] used hidden Markov models to generate Bach chorale harmonizations and is one of the first studies to evaluate model performance quantitatively using cross-entropy on held-out data. They introduce the *JSB Chorales* dataset which has since become a standard benchmark routinely used to evaluate the performance of generative models on polyphonic music modelling [4, 33, 2, 21, 41]. However, *JSB Chorales* quantizes time to eighth notes, distorting 2816 notes (2.85% of the corpus). In contrast, BachBot eliminates this problem with $2\times$ the time resolution (distorting no notes). Unfortunately, the higher resolution time quantization of Bach-

Bot's data as well as BachBot's sequential encoding format make direct comparison of cross-entropies against studies using this dataset difficult. On this dataset, the current state-of-the-art (as measured by cross-entropy validation loss) by Goel and Vohra [20] uses a deep belief network (DBN) which uses a LSTM to propagate temporal dynamics. While BachBot also utilizes a LSTM for capturing long range dependencies, BachBot uses a softmax distribution rather than a DBN to parameterize the probability distribution and hence does not require Monte Carlo sampling at each time step of training and inference.

A recent approach developed concurrent to BachBot was by Hadjeres and Pachet [23]. Their approach also uses an encoding which accounts for note articulations and fermatas and is similarly capable of harmonization under arbitrary constraints (e.g. a given Alto and Tenor part). However, their model utilizes LSTMs to summarize both past and future context within ± 16 time steps, limiting context to a temporally local region and inhibiting the learning of long-term structures such as motifs. Since future context is not always available, to generate samples the authors first randomly initialize a predetermined number of time steps followed by multiple iterations of MCMC. In contrast, BachBot's ancestral sampling method requires only a single forward pass and does not require the number of timestamps in the sample to be known in advance. The authors also evaluate their model using an online discrimination test, but on a smaller participant pool of 1272.

2. THE BACHBOT SYSTEM

2.1 Corpus Construction and Preprocessing

We took the full set of Bach chorales in MusicXML format as provided by Cuthbert and Ariza [10]. Following prior work [31, 14, 16, 17] preprocessing transposed all scores to C-major / A-minor and quantized time into sixteenth notes. Time quantization at this resolution does not distort any notes in the corpus.

2.2 Sequential Encoding of Polyphonic Music Scores

We encode the scores into sequences of tokens amenable for sequential processing by recurrent neural networks (RNNs). We limit the symbolic representation to pitch and rhythm. This is consistent with previous work [4, 33] and the practice of music theoretic pedagogy. Unlike some prior work [15, 14, 1], we avoid explicitly encoding music-theoretic concepts such as motifs, phrases, and chords / inversions, instead tasking the model to learn musically meaningful features with minimal prior knowledge (see section 3.4).

Our encoding represents polyphonic scores with sixteenth-note *frames*, encoding duration implicitly by the number of frames processed. Such an encoding requires the network to leverage memory to account for longer durations notes, a counting and timing task which LSTM is known to be capable of [19]. Consecutive frames are separated by a unique delimiter (`|||` in fig. 1).

Within each frame, we represent individual notes rather than entire chords, reducing the vocabulary size from $O(128^4)$ down to $O(128)$. Prior work modeling characters versus words in language modeling tasks suggests that this has negligible impact [22]. Each frame consists of four (Soprano, Alto, Tenor, and Bass) $\langle \texttt{Pitch}, \texttt{Tie} \rangle$ tuples where $\texttt{Pitch} \in \{0, 1, \cdots, 127\}$ represents the MIDI pitch of a note and $\texttt{Tie} \in \{\texttt{True}, \texttt{False}\}$ distinguishes whether a note is tied with a note at the same pitch from the previous frame or is articulated at the current timestep. We *order notes within a frame in descending MIDI pitch and neglects crossing voices*; potential consequences of doing so are discussed in section 3.2.

For each score, a unique `START` symbol and `END` symbol are added. This enables initialization of the trained model prior to ancestral sampling of a token sequence by providing a `START` token and also allows us to determine when a sampled composition ends. In addition, our encoding also includes fermatas (represented by `(.)`), which Bach used to denote ends of phrases. Significantly, we found that adding this additional notation to the input resulted in more realistic phrase lengths in generated output.

2.3 Model Architecture, Training, and Sampling

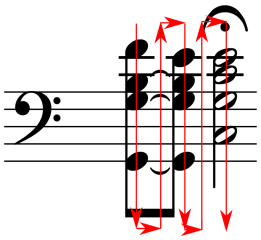
We use a RNN with LSTM memory cells and the following hyperparameters:

1. `num_layers` – the number of memory cell layers
2. `rnn_size` – the number of hidden units per memory cell (*i.e.* hidden state dimension)
3. `wordvec` – dimension of vector embeddings
4. `seq_length` – number of frames before truncating back-propagation through time (BPTT) gradient
5. `dropout` – the dropout probability

Our model first embeds the inputs $\boldsymbol{x}_t$ into a `wordvec`-dimensional vector-space, compressing the dimensionality down from $|V| \approx 140$ to `wordvec` dimensions. Next, `num_layers` layers of memory cells followed by batch normalization [28] and dropout [26] with dropout probability `dropout` are stacked. The outputs $\boldsymbol{y}_t^{(\texttt{num\_layers})}$ are followed by a fully-connected layer mapping to $|V| = 108$ units, which are passed through a softmax to yield a predictive distribution $P(\boldsymbol{x}_{t+1}|\boldsymbol{h}_{t-1}, \boldsymbol{x}_t)$: the probability distribution over the next token $\boldsymbol{x}_{t+1}$ given the current token $\boldsymbol{x}_t$ and the previous RNN memory cell state $\boldsymbol{h}_{t-1}$.

Models were trained using the Adam optimizer [29] with a minibatch size of 50 and an initial learning rate of 2×10^{-3} decayed by 0.5 every 5 epochs. The back-propagation through time gradients were clipped at ± 5.0 [32] and truncated after `seq_length` frames.

We minimize cross-entropy loss between the predicted distributions $P(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{h}_{t-1})$ and the actual target distribution $\delta_{\boldsymbol{x}_{t+1}}$. During training, the correct token $\boldsymbol{x}_{t+1}$ is treated as the model output even if the most likely prediction $\operatorname{argmax} P(\boldsymbol{x}_{t+1}|\boldsymbol{h}_t, \boldsymbol{x}_t)$ differs. Williams and Zipser



(a) Three musical chords in traditional music notation. The red arrows indicate the order in which notes are sequentially encoded.

```
START
(65, False)
(59, False)
(55, False)
(43, False)
|||
(64, False)
(59, True)
(55, True)
(43, True)
|||
(.)
(64, False)
(60, False)
(55, False)
(48, False)
|||
END
```

(b) A corresponding sequential encoding of the three chords in an eighth-note time-quantization (for illustration, broken over three columns). Each line within a column corresponds to an individual token in the encoded sequence. `|||` delimit frames and `(.)` indicate a fermata is present within the corresponding frame.

Figure 1: Example encoding of three musical chords ending with a fermata ("pause") chord.

[40] refers to this as *teacher forcing*, which is performed to aid convergence because the model's predictions may not be reliable early in training. During inference, we perform ancestral sampling and reuse the actual token $\hat{\boldsymbol{x}}_t$ sampled from $P(\boldsymbol{x}_t|\boldsymbol{h}_{t-1}, \boldsymbol{x}_{t-1})$ to compute $P(\boldsymbol{x}_{t+1}|\boldsymbol{h}_t, \boldsymbol{x}_t)$ for sampling $\hat{\boldsymbol{x}}_{t+1}$. Unlike MCMC, which requires running multiple iterations to obtain a single sample, ancestral sampling requires only a single forward pass.

2.4 Harmonization with Greedy 1-best Search

Chorale harmonization involves providing accompaniment parts to an existing melody. This is a musical task with ecological validity undertaken by many composers including Bach himself. Many of Bach's chorales are harmonizations by Bach of pre-existing melodies (not by Bach) and certain melodies (by Bach or otherwise) form the basis of multiple chorales with different harmonizations.

We extend this harmonization task to the completion of chorales for a wider number and type of given parts. Let $\boldsymbol{x}_{(1:T)}$ be a sequence of tokens representing an encoded musical score, $\alpha \subset \{1, 2, \ldots, T\}$ a multi-index, and suppose $\widehat{\boldsymbol{x}}_\alpha$ correspond to some fixed token values to be harmonized (e.g. a provided Soprano line).

We are interested in solving the following optimization:

$$\boldsymbol{x}^*_{(1:T)} = \underset{\boldsymbol{x}_{(1:T)}}{\operatorname{argmax}} P(\boldsymbol{x}_{(1:T)}|\boldsymbol{x}_\alpha = \widehat{\boldsymbol{x}}_\alpha) \tag{1}$$

First, any proposed solution $\tilde{\boldsymbol{x}}_{1:T}$ must satisfy $\tilde{\boldsymbol{x}}_\alpha = \widehat{\boldsymbol{x}}_\alpha$, so the decision variables are $\tilde{\boldsymbol{x}}_{(1:T)\setminus\alpha}$. Hinton and Sejnowski [25] refer to this constraint as "clamping" the generative model. We propose a simple greedy strategy for choosing $\tilde{\boldsymbol{x}}_{(1:T)\setminus\alpha}$:

$$\tilde{\boldsymbol{x}}_t = \begin{cases} \widehat{\boldsymbol{x}}_t & \text{if } t \in \alpha \\ \operatorname{argmax}_{\boldsymbol{x}_t} P(\boldsymbol{x}_t|\tilde{\boldsymbol{x}}_{1:t-1}) & \text{otherwise} \end{cases} \tag{2}$$

where the tilde on the previous tokens $\tilde{\boldsymbol{x}}_{1:t-1}$ indicate that they are equal to the actual previous argmax choices. This corresponds to a greedy 1-best search at each time t without any accounting of future constraints (e.g. $\boldsymbol{x}_\tau$ if $\tau > t$ and $\tau \in \alpha$). This is sub-optimal, and we leave more sophisticated search strategies such as beam search [30] for future work.

3. EXPERIMENTS

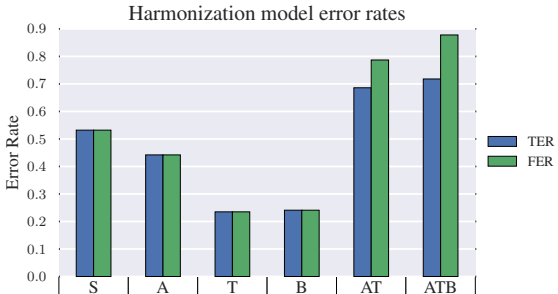
3.1 Sequence Modelling

With the BachBot model, we performed a grid search through the parameter grid in table 1 and found `num_layers` $= 3$, `rnn_size` $= 256$, `wordvec` $= 32$, `seq_length` $= 128$ `dropout` $= 0.3$ achieves the lowest cross-entropy loss of 0.477 bits on a 10% held-out validation corpus.

Parameter	Values Searched
`num_layers`	$\{1, 2, 3, 4\}$
`rnn_size`	$\{128, 256, 384, 512\}$
`wordvec`	$\{16, 32, 64\}$
`seq_length`	$\{64, 128, 256\}$
`dropout`	$\{0.0, 0.1, 0.2, 0.3, 0.4, 0.5\}$

Table 1: The grid of hyperparameters searched over while optimizing RNN structure

3.2 Harmonization



	S	A	T	B	AT	ATB
TER	0.532	0.442	0.235	0.241	0.686	0.718
FER	0.532	0.442	0.235	0.241	0.787	0.878

Figure 2: Token error rates (TER) and frame error rates (FER) for various harmonization tasks

For the parts to harmonize (i.e. $\boldsymbol{x}_{(1:T)\setminus\alpha}$), we considered the following test cases:

1. One part: Soprano (S), Alto (A), Tenor (T), or Bass (B).

2. The inner parts (AT). Completion of the inner parts corresponds to a musically-valid exercise common in Baroque composition (including some Bach chorales) where only the outer voices are specified (with or without figured bass to indicate the chord types).

3. All parts except Soprano (ATB): the most common form of *harmonization* exercise.

It is widely accepted that these tasks successively increase in terms of difficulty [11].

We deleted the different subsets of parts from a validation corpus and used eq. (2) to fill in the missing parts. Our model's error rates for predicting individual tokens (token error rate, TER, % of errors in individual token predictions) as well as all tokens within frames (frame error rate, FER, % of errors in frame predictions where any token prediction errors within a frame counts as a frame error) are reported in fig. 2.

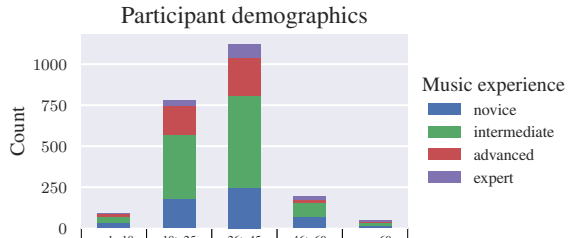
Surprisingly, error rates were higher for S/A than for T/B. One possible explanation for this result is our design decision in section 2.2 to order notes within a frame in SATB order. As a result, the model must predict the Soprano part for each frame without any knowledge of the other parts. When predicting the Bass part, however, it has already seen all of the other parts and can leverage this harmonic context. To assess this idea, we propose as future work an investigation of different part orderings in the encoding.

3.3 Musical Discrimination Test

To measure BachBot's success in this task, we developed a publicly accessible musical discrimination test at `bachbot.com`. Unlike prior studies which leverage paid services like Amazon MTurk for human feedback [35], we offered no such incentive and promoted the study only through social media.

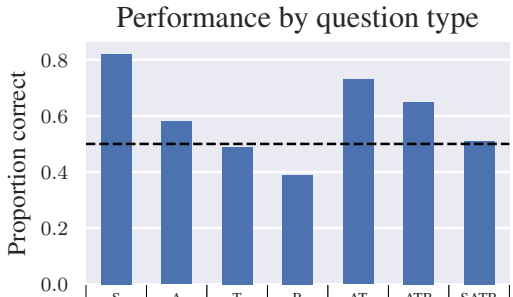
Participants were first surveyed for their age group and prior music experience (fig. 3a). Next, they are presented five discrimination tasks which presented two audio tracks (an original Bach composition and a synthetic composition by BachBot) and ask them to identify the Bach original. Each audio track contains an entire composition from start to end. The music score for the audio was not provided. Participants were granted an unlimited amount of time and allowed to replay each track an arbitrary number of times. Participants could only see the next question after submitting the current one and were not allowed to modify their responses after submitting.

The five questions comprised of three harmonizations (S/A/T/B, one AT, one ATB), and two original compositions. To construct the questions, harmonizations were paired along with the original Bach chorales the fixed parts were taken from. No such direct comparison is possible for the SATB case, so these synthetic compositions were paired with a randomly selected Bach chorale in a somewhat different comparative listening task. Harmonizations



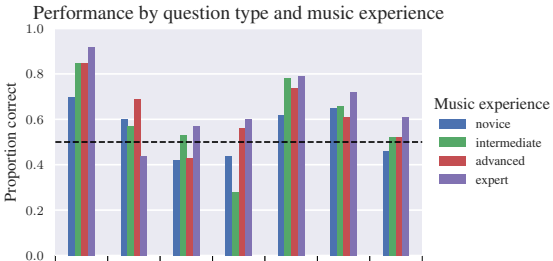
	under18	18to25	26to45	46to60	over60
novice	34	181	244	66	16
intermediate	36	387	565	85	18
advanced	17	176	233	23	5
expert	5	34	81	21	9

(a) Demographics of respondents; self-reported music experience defined as follows — *Novice*: casual listener, *Intermediate*: plays an instrument, *Advanced*: formally studied music composition, *Expert*: music teacher/researcher.



	S	A	T	B	AT	ATB	SATB
Proportion	0.82	0.58	0.49	0.39	0.73	0.65	0.51

(b) Proportion of responses correctly discriminating BachBot from Bach for different question types. The SATB column shows that BachBot's generated compositions can be differentiated from Bach only 1% better than random guessing.



	S	A	T	B	AT	ATB	SATB
novice	0.7	0.6	0.42	0.44	0.62	0.65	0.46
intermediate	0.85	0.57	0.53	0.28	0.78	0.66	0.52
advanced	0.85	0.69	0.43	0.56	0.74	0.61	0.52
expert	0.92	0.44	0.57	0.6	0.79	0.72	0.61

(c) Figure 3b segmented by self-reported music experience. As expected, more experienced listeners generally produced more correct responses, though not for the 'B' condition.

Figure 3: Results collected from a web-based musical discrimination test.

Figure 4: Activation profiles suggesting that neurons have specialized to become detectors of musically relevant features. *Layer 1, neuron 64*: strongly correlates with the use of dominant seventh chords in the main, tonic key (C major, originally D major). These are the main non-triadic harmony, are strongly key defining, and have a important function in the harmonic closure of phrases in this style. *Layer 1, neuron 151*: fires with the equivalent dominant seventh chord for the two cadences in the relative minor (a minor, originally b minor) that end phrases 2 and 4. These are the only two appearances in the chorale of the pitch G# which is foreign to C major, and strongly key defining in a minor.

were synthesized by extracting part(s) from a randomly selected Bach chorale and filling in the remaining parts of the composition using the method previously described in section 2.4. Original compositions (questions labelled SATB) were generated by providing a `START` symbol followed by ancestral sampling as previously described in section 2.3 until an `END` symbol is reached. The final audio provided in the questions were obtained by rendering the compositions using the Piano instrument from the Fluid R3 GM SoundFont.

We only considered the first response per IP address of participants who had played both choices in every question at least once and completed all five questions. This totaled 2336 participants at the time of writing, making our study one of the largest subjective listening evaluation of an automatic composition system to date.

Figure 3b shows the performance of BachBot on various question types. The SATB column shows that, for the novel synthetic compositions, participants on average successfully discriminated Bach from BachBot only 51%: *average human listeners could only perform* 1% *better than random guessing*. To assess statistical significance, we choose significance level $\alpha = 0.05$ and conducted a one-tailed binomial test (446 successes in 874 trials) to find that the probability of a discrimination rate higher than 51% has p-value $0.282 > \alpha$. Thus, we conclude that there does not exist sufficient evidence that the discrimination rate between Bach and BachBot is significantly different (at $\alpha = 0.05$) than the rate achieved by random guessing random guessing .

The weaker performance of BachBot's outputs on most harmonization questions (fig. 3b other than SATB) compared to automatic composition questions (SATB) is counterintuitive: one would expect the provided parts to aid the model in creating more Bach-like music. This result may be explained by the shortcomings of our greedy 1-best harmonization method (discussed above) and/or by the possible benefit of consistent origins, with all-Bach and all-BachBot being preferred over hybrid solutions.

Across the S/A/T/B and AT/ATB conditions, the results vary significantly. The ease of discrimination appears to correlate with the position in the texture from highest (S, easiest) to lowest (B, hardest). This may be due to the S part's importance in carrying the melody in chorale style, or (more likely) due once again to the BachBot's lower error rates for completing bass parts as compared with other parts (fig. 2), which in turn is probably due to the sequential encoding (fig. 1) of bass notes last within each frame, giving it a harmonic context to work with. Another possibility is that most listeners focus more on the top melody, neglecting the bass part and any potential deviations there. In any case, the relatively poor performance of expert listeners for the B-only condition (see fig. 3c) is noteworthy, and not explained by any aspect of the process.

3.4 Do Neurons Specialize to Music-Theoretic Concepts?

Research in convolutional networks has shown that neurons within computer vision models specialize to detect high-level visual features [42]. Similarly, convolutional networks trained on audio spectrograms have been shown to possess neurons which detect high-level aural features [5]. Following these results, one might expect the BachBot model to possess neurons which detect features within symbolic music which have music theoretic relevance.

To investigate this further, one could look at the activations over time of individual neurons within the LSTM memory cells to see if neuron activity correlates with recognized musical processes. An informal analysis suggests that while some neurons are ambiguous to interpretation, other neurons correlate significantly with recognized music-theoretic objects, particularly chords (see fig. 4). To our knowledge, *this is the first reported evidence for an LSTM optimized for automatic composition learning music-theoretic concepts without explicit prior information*. This invites a follow-up study testing the statistical significance of these observations.

4. DISCUSSION

The data generated by `bachbot.com` shows that subjects distinguished BachBot from Bach only 51% of the time, suggesting that BachBot successfully composes and completes music that cannot be distinguished from Bach significantly above the chance level. Additionally, BachBot's design involves no explicit encoding of musical parameters beyond the notation, so the results reflects its ability to acquire music knowledge independently from data.

As discussed, the higher time resolution of our custom encoding scheme enabled the model to learn about Bach's use of sixteenth notes, which is not possible for models trained on *JSB Chorales*. Unfortunately, this improved encoding means that we are unable to compare quantitative performance metrics such as log likelihood against other literature values reported for polyphonic modeling on the *JSB Chorales* [1] dataset.

Using this sequential encoding scheme, we train a deep LSTM sequential prediction model and discover that it learns music theoretic concepts without prior knowledge or explicit supervision. We then propose a method to utilize the sequential prediction model for harmonization tasks. We acknowledge that our method is not ideal and discuss better alternatives in future work. Our harmonization results reveal that this issue is significant and should be a priority for any follow-up work.

Finally, we leveraged our model to generate harmonizations as well as novel compositions and used the generated music in a web-based music discrimination test. Our results here confirm the success of our project.

While many opportunities for extension are highlighted, we conclude that our stated research aims have been reached. In other words, generating stylistically successful Bach chorales is now a more closed (as a result of BachBot) than open problem.

5. CONCLUSION

In this paper, we:

- introduce a sequential encoding scheme for music which achieves time-resolution $2\times$ that of the commonly used *JSB Chorales* [1] dataset.
- performed the largest (to the best of our knowledge at time of publication) musical discrimination test of an automatic composition system, which demonstrated that high quality data can be collected from voluntary internet surveys.
- demonstrate that a deep LSTM sequential prediction model trained on our encoding scheme is capable of composing music that can be distinguished only 1% better than random guessing, a statistically insignificant difference
- provide the first evidence that neurons in the LSTM model appear to model common music-theoretic concepts without prior knowledge or supervision.

In addition, we have open sourced the code for BachBot [1] as well as our music discrimination test framework [2]. The Magenta project of Google Brain has recently implemented the BachBot model for their polyphonic RNN model [3].

6. REFERENCES

[1] Moray Allan and Christopher KI Williams. allan2005. *Advances in Neural Information Processing Systems*, 17:25–32, 2005.

[2] Justin Bayer, Christian Osendorfer, Daniela Korhammer, Nutan Chen, Sebastian Urban, and Patrick van der Smagt. On fast dropout and its applicability to recurrent networks. *arXiv preprint arXiv:1311.0701*, 2013.

[3] Matthew I Bellgard and Chi-Ping Tsang. Harmonizing music the boltzmann way. *Connection Science*, 6 (2-3):281–297, 1994.

[4] Nicolas Boulanger-Lewandowski, Pascal Vincent, and Yoshua Bengio. Modeling Temporal Dependencies in High-Dimensional Sequences: Application to Polyphonic Music Generation and Transcription. *Proc. of the 29th International Conference on Machine Learning (ICML-12)*, (Cd):1159–1166, 2012.

[1] `https://github.com/feynmanliang/bachbot`

[2] `https://github.com/feynmanliang/subjective-evaluation-server` and `https://github.com/feynmanliang/subjective-evaluation-client`

[3] `https://github.com/tensorflow/magenta/tree/master/magenta/models/polyphony_rnn`

[5] Keunwoo Choi, George Fazekas, Mark Sandler, and Jeonghee Kim. Auralisation of deep convolutional neural networks: Listening to learned features. In *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR*, pages 26–30, 2015.

[6] Tom Collins, Robin Laney, Alistair Willis, and Paul H Garthwaite. Developing and evaluating computational models of musical style. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 30(01):16–43, 2016.

[7] David Cope. Experiments in music intelligence. In *Proc. of the International Computer Music Conference*, 1987.

[8] David Cope. Computer modeling of musical intelligence in emi. *Computer Music Journal*, 16(2):69–83, 1992.

[9] Pedro P Cruz-Alcázar and Enrique Vidal-Ruiz. Learning regular grammars to model musical style: Comparing different coding schemes. In *International Colloquium on Grammatical Inference*, pages 211–222. Springer, 1998.

[10] Michael Scott Cuthbert and Christopher Ariza. music21: A toolkit for computer-aided musicology and symbolic music data. 2010.

[11] James Denny. *The Oxford school harmony course*, volume 1. Oxford University Press, 1960.

[12] Kemal Ebcioğlu. An expert system for harmonizing four-part chorales. *Computer Music Journal*, 12(3): 43–51, 1988.

[13] D. Eck and J. Schmidhuber. Finding temporal structure in music: Blues improvisation with LSTM recurrent networks. *Neural Networks for Signal Processing - Proc. of the IEEE Workshop*, 2002-Janua: 747–756, 2002. ISSN 0780376161. doi: 10.1109/NNSP.2002.1030094.

[14] Douglas Eck and Jürgen Schmidhuber. A 1st Look at Music Composition using LSTM Recurrent Neural Networks. *Idsia*, 2002. URL `http://www.idsia.ch/{~}juergen/blues/IDSIA-07-02.pdf`.

[15] Johannes Feulner and Dominik Hörnel. Melonet: Neural networks that learn harmony-based melodic variations. In *Proc. of the International Computer Music Conference*, pages 121–121. INTERNATIONAL COMPUTER MUSIC ACCOCIATION, 1994.

[16] Judy A Franklin. Recurrent neural networks and pitch representations for music tasks. In *FLAIRS Conference*, pages 33–37, 2004.

[17] Judy A Franklin. Jazz melody generation from recurrent network learning of several human melodies. In *FLAIRS Conference*, pages 57–62, 2005.

[18] Judy A Franklin. Recurrent neural networks for music computation. *INFORMS Journal on Computing*, 18(3):321–338, 2006.

[19] Felix A Gers, Nicol N Schraudolph, and Jürgen Schmidhuber. Learning precise timing with lstm recurrent networks. *Journal of machine learning research*, 3(Aug):115–143, 2002.

[20] Kratarth Goel and Raunaq Vohra. Learning temporal dependencies in data using a dbn-blstm. *arXiv preprint arXiv:1412.6093*, 2014.

[21] Kratarth Goel, Raunaq Vohra, and JK Sahoo. Polyphonic music generation by modeling temporal dependencies using a rnn-dbn. In *International Conference on Artificial Neural Networks*, pages 217–224. Springer, 2014.

[22] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[23] Gaëtan Hadjeres and François Pachet. Deepbach: a steerable model for bach chorales generation. *arXiv preprint arXiv:1612.01010*, 2016.

[24] Hermann Hild, Johannes Feulner, and Wolfram Menzel. Harmonet: A neural net for harmonizing chorales in the style of js bach. In *NIPS*, pages 267–274, 1991.

[25] Geoffrey E Hinton and Terrence J Sejnowski. Learning and releaming in boltzmann machines. *Parallel distributed processing: Explorations in the microstructure of cognition*, 1:282–317, 1986.

[26] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[27] Dominik Hörnel. Melonet i: Neural nets for inventing baroque-style chorale variations. In *NIPS*, pages 887–893, 1997.

[28] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[29] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[30] Xunying Liu, Yongqiang Wang, Xie Chen, Mark JF Gales, and Philip C Woodland. Efficient lattice rescoring using recurrent neural network language models. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4908–4912. IEEE, 2014.

[31] Michael C Mozer. Neural network music composition by prediction: Exploring the benefits of psychoacoustic constraints and multi-scale processing. *Connection Science*, 6(2-3):247–280, 1994.

[32] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. *Proc. of The 30th International Conference on Machine Learning*, (2):1310–1318, 2012. ISSN 1045-9227. doi: 10.1109/72.279181. URL `http://jmlr.org/proceedings/papers/v28/pascanu13.pdf`.

[33] Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. How to construct deep recurrent neural networks. *arXiv preprint arXiv:1312.6026*, 2013.

[34] Marcus Pearce and Geraint Wiggins. Towards a framework for the evaluation of machine compositions. In *Proc. of the AISB'01 Symp. on Artificial Intelligence and Creativity in the Arts and Sciences*, pages 22–32. Citeseer, 2001.

[35] Donya Quick. *Kulitta: A Framework for Automated Music Composition*. PhD thesis, YALE UNIVERSITY, 2014.

[36] Randall R Spangler, Rodney M Goodman, and Jim Hawkins. Bach in a box-real-time harmony. 1998.

[37] Peter Todd. A sequential network design for musical applications. In *Proc. of the 1988 connectionist models summer school*, pages 76–84, 1988.

[38] Peter M Todd. A connectionist approach to algorithmic composition. *Computer Music Journal*, 13(4): 27–43, 1989.

[39] Chi Ping Tsang and Melanie Aitken. Harmonizing music as a discipline in contraint logic programming. In *Proc. of the International Computer Music Conference*, pages 61–61. INTERNATIONAL COMPUTER MUSIC ACCOCIATION, 1991.

[40] Ronald J Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

[41] Wojciech Zaremba. An empirical exploration of recurrent network architectures. 2015.

[42] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Rob Fergus. Deconvolutional networks. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2528–2535. IEEE, 2010.