Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI-22)
Special Track on AI, the Arts and Creativity

Tradformer: A Transformer Model of Traditional Music Transcriptions

Luca Casini[1] , **Bob L. T. Sturm**[2]
[1]University of Bologna, Department of Computer Science and Engineering. Bologna, Italy
[2]Kungliga Tekniska Högskola, Stockholm, Sweden
luca.casini7@unibo.it, bobs@kth.se

Abstract

We explore the transformer neural network architecture for modeling music, specifically Irish and Swedish traditional dance music. Given the repetitive structures of these kinds of music, the transformer should be as successful with fewer parameters and complexity as the hitherto most successful model, a vanilla long short-term memory network. We find that achieving good performance with the transformer is not straightforward, and careful consideration is needed for the sampling strategy, evaluating intermediate outputs in relation to engineering choices, and finally analyzing what the model learns. We discuss these points with several illustrations, providing reusable insights for engineering other music generation systems. We also report the high performance of our final transformer model in a competition of music generation systems focused on a type of Swedish dance.

1 Introduction

Since the introduction of transformers [Vaswani *et al.*, 2017], the state of the art of numerous applications of machine learning has advanced, from natural language processing [Radford *et al.*, 2019; Devlin *et al.*, 2018] to computer vision [Dosovitskiy *et al.*, 2020]. At the core of a transformer is *attention*: a dynamic and highly parallelizable mechanism by which the model predicts output from previous input. A transformer can potentially learn dependencies over very long ranges, addressing one of the biggest limiting factors of recurrent neural networks like long short-term memory (LSTM) [Casini *et al.*, 2018]. For music sequences exhibiting repetition, a transformer can learn to refer back to what it has already generated, attending to particular material to make inferences. The ability to repeat material over long time scales is seen in the outputs of *Music Transformer* [Huang *et al.*, 2018] and *Musenet* [Payne, 2019]. However, transformers have their limits, such as a maximum sequence length and a quadratic memory requirement that scales with it.

The application of attention to modeling transcriptions of traditional music has yet to be thoroughly explored. All entries submitted to *The Ai Music Generation Challenge 2020*[1] (focusing on Irish double jigs) used either Markov modeling or LSTM [Sturm and Maruri-Aguilar, 2021]. The winning entries were generated by the LSTM-based system *folk-rnn (v2)* [Sturm *et al.*, 2016]. Brawen (2019) explores tuning the language transformer model GPT-2 on text-based traditional music data; but this model has about 20 times the number of parameters of *folk-rnn* and little demonstrable gain in quality.

Given attention and its compatibility with the kinds of structures present in Irish and Swedish traditional dance music, a transformer model should excel in generating plausible new tunes in these styles with fewer parameters than the LSTM *folk-rnn*. Applying the transformer to this task however, was not straightforward. Certain aspects of development had a large influence on performance. To develop a successful transformer model, we performed several iterations of informal comparison tests with *folk-rnn* models as a benchmark, and also developed visualizations of the internal structure of the transformer, including its use of attention on forming predictions. The final model, which we name *Tradformer*, has about one-fifth the parameters of *folk-rnn* and achieves results of similar or better quality for some forms of Irish traditional music. We used transfer learning to adapt *Tradformer* to a style of Swedish traditional dance, called slängpolska, which is the focus of *The Ai Music Generation Challenge 2021*.[2] Competing against five others, *Tradformer* has produced tunes rated the most favorable by five human judges.

In the following, we review the transformer architecture and then present the details of our model. Section 4 discusses numerous decisions we made in engineering *Tradformer*, including sampling and visualizing the internal components of the model as training proceeds. Section 5 presents how we have used transfer learning to tune *Tradformer* to adapt to a type of Swedish music. We discuss our results in Sec. 6.

2 Review of a Transformer Model

Let $\mathcal{V} = \{v_i : i \in \mathcal{I}\}$ be a vocabulary indexed by the set $\mathcal{I}$, and a length-N sequence $s = (i_t \in \mathcal{I} : t \in [0, N))$ where t is the index into the sequence. The output of a transformer is a categorical probability distribution over $\mathcal{I}$ conditioned on s,

[1]https://boblsturm.github.io/aimusic2020/
[2]https://github.com/boblsturm/aimusicgenerationchallenge2021

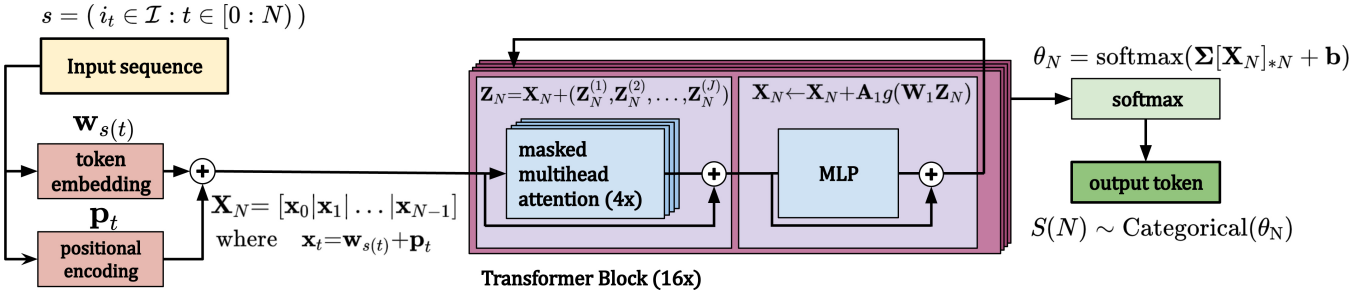

Figure 1: Illustration of a transformer architecture processing a length-N sequence s to produce a categorical probability distribution for the next sequence element $S(N)$.

from which the Nth element $S(N)$ can be inferred. Figure 1 depicts the basic transformer, which operates as follows.

Each element of s is represented by an embedding vector in a d_e-dimensional real space, summed with a vector encoding its position in the sequence [Vaswani *et al.*, 2017]. More precisely, element t of s is encoded as $\mathbf{w}_{s(t)} \in \mathbb{R}^{d_e}$, and its position is encoded as $\mathbf{p}_t \in \mathbb{R}^{d_e}$. Let $\mathbf{X}_N = [\mathbf{x}_0|\mathbf{x}_1|\ldots|\mathbf{x}_{N-1}]$, where $\mathbf{x}_t = \mathbf{w}_{s(t)} + \mathbf{p}_t$. This is processed by the first layer of J attention heads, each operating in a subspace of $\mathbb{R}^{d_e}$ of dimension $d_h = d_e/J$ (where d_e is an integer multiple of J). The jth attention head produces query, key and value representations via: $\mathbf{Q}_N^{(j)} = \mathbf{W}_j^Q \mathbf{X}_N$, $\mathbf{K}_N^{(j)} = \mathbf{W}_j^K \mathbf{X}_N$, and $\mathbf{V}_N^{(j)} = \mathbf{W}_j^V \mathbf{X}_N$, where $\mathbf{W}_j^Q, \mathbf{W}_j^K, \mathbf{W}_j^V \in \mathbb{R}^{d_h \times d_e}$. The jth attention head then forms the scaled inner-product matrix $\mathbf{S}_N^{(j)} = (d_h)^{-1/2} [\mathbf{Q}_N^{(j)}]^T \mathbf{K}_N^{(j)}$, compresses each column of $\mathbf{S}_N^{(j)}$ using a softmax operation to form $\hat{\mathbf{S}}_N^{(j)} \in \mathbb{R}^{N \times N}$, and outputs $\mathbf{Z}_N^{(j)} = \mathbf{V}_N^{(j)} \hat{\mathbf{S}}_N^{(j)} \in \mathbb{R}^{d_h \times N}$. The outputs of all J attention heads are concatenated $\mathbf{Z}_N = \mathbf{X}_N + (\mathbf{Z}_N^{(1)}, \mathbf{Z}_N^{(2)}, \ldots, \mathbf{Z}_N^{(J)}) \in \mathbb{R}^{d_e \times N}$, and then passed through a non-linearity $\mathbf{Z}'_N = g(\mathbf{W}_1 \mathbf{Z}_N)$, where $\mathbf{W}_1 \in \mathbb{R}^{4d_e \times d_e}$. Finally, the output of the first layer is $\mathbf{X}_N \leftarrow \mathbf{X}_N + \mathbf{A}_1 \mathbf{Z}'_N$ where $\mathbf{A}_1 \in \mathbb{R}^{d_e \times 4d_e}$. The following attention layer performs the same operation as the first layer, but on the updated $\mathbf{X}_N$ and with different parameters. After L attention layers, the final layer takes the last column of $\mathbf{X}_N$ and computes the parameters of the distribution over $\mathcal{I}$ by $\theta_N = \mathrm{softmax}(\mathbf{\Sigma}[\mathbf{X}_N]_{*N} + \mathbf{b})$ where $\mathbf{\Sigma} \in \mathbb{R}^{|\mathcal{I}| \times d_e}$ and $\mathbf{b} \in \mathbb{R}^{|\mathcal{I}|}$.

Training a transformer entails learning the embedding vectors of the vocabulary, the linear transformations of the query, key, and value in each attention head in each layer, the linear transformations of the concatenations of the results of attention, the weights between each hidden layer, and finally the parameters of the softmax layer. The standard loss function is cross-entropy.

3 Tradformer Architecture and Training

We aim to build a transformer that performs as well as the LSTM *folk-rnn (v2)* [Sturm *et al.*, 2016], and so use the same data and representation.[3] This dataset comes from thesession.org, and contains 23,636 transcriptions, mainly of traditional Irish dance music, expressed with a vocabulary of 137 tokens representing meter, mode, pitch, etc. We limit the maximum length to 256, removing only 2,362 sequences.

We make the dimension of the vocabulary embedding $d_e = 128$. Using a dimension larger than the vocabulary adds complexity and training time, and we found no benefit through human evaluation (described more in Sec. 4). We also found worse results using a dimension of 64 (powers of 2 are not necessary but often they allow the hardware to work faster). The positional encoding of *Tradformer* is that given in [Vaswani *et al.*, 2017], which is based on a combination of sinusoids. This can be learned as well, but our evaluation showed it to work slightly worse. *Tradformer* has $L = 16$ layers each with $J = 4$ attention heads. This was determined by testing numbers of layers ranging from 4 to 16, and numbers of attention heads from 1 to 8.

We experimented with different batch sizes, learning rates and numbers of epochs. The final learning rate is 5.0^{-4}. The batch size was set to 64 from considerations of our computational hardware. We used in total 50 epochs. These parameters however did not seem to have a major impact on model convergence. When *Tradformer* showed small to no improvement in its validation loss during training, we could still detect improvement in the quality of generated tunes. An explanation could be that, looking at the loss function, a wrong token is still wrong even if it is musically plausible. As training goes by, the average number of mistakes could be the same but the quality of those mistakes could be improving from a music theory standpoint.

For sampling, *Tradformer* employs a combination of beam search and nucleus sampling similar to that proposed in [Shaham and Levy, 2021]. *Tradformer* starts with top-p sampling (only looks at the most likely token up to p percent of the probability mass) and then uses at most k of those as branches to search a tree of depth of maximum depth D. This results in a sample space of at most k^D token sequences with a probability distribution given by the softmax of the sum of the logits of the component tokens. This increases computational cost, but the increase in output quality was clear. *Tradformer* sets $D = 3$, $p = 0.99$ and $k = 3$.

We find by human evaluation that *Tradformer* performs as well as *folk-rnn (v2)* in generating tunes in 6/8 and 4/4 meter. The appendix contains the scores and comments for some of

[3] See `data_v2` in https://github.com/IraKorshunova/folk-rnn

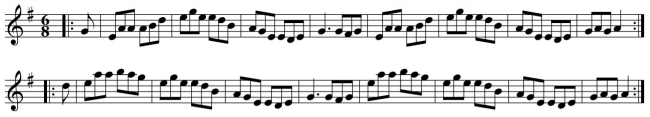

Figure 2: *Tradformer* output given a seed specifying 6/8 meter, transposed to a characteristic mode for Irish music.

these comparisons[4]. Figure 2 shows one example generated by *Tradformer*. The tune has a very common bipartite structure, with each part demarcated by repeat signs. The parts are very similar and only differ in their first bars – a common structure in Irish traditional dance music. The only odd part is that the ending of each section does not resolve.

4 Engineering Considerations

This section discusses a variety of considerations that guided the engineering of *Tradformer*.

4.1 Human Evaluation

Many design choices were dictated not only by measuring a difference in loss, but also the quality of model outputs by human evaluation. Periodically, we had *Tradformer* and *folk-rnn (v2)* each generate 10 tunes in a given meter, ordered them randomly without any identification, and then had a musician competent in Irish traditional music rate each tune as good or bad. This provided a more complete picture of the musical quality of the two models, and revealed when the two models were generating material at a similar level of quality. It was an inexpensive and effective way of gauging progress.

4.2 Sampling

The sampling strategy of *Tradformer* is extremely important. We found the biggest improvement in the quality of the generated music came from replacing a naive approach with a more sophisticated one based on beam search. Our early models were using a naive sampling approach, where single tokens were drawn from a categorical distribution parametrized with the softmax probabilities. The outputs contained counting errors and drifting melodies, and were rated very poor against melodies generated by *folk-rnn*.

Lowering the softmax temperature was slightly helpful in preventing such problems, but created too much repetition. Furthermore, temperature is applied equally even when the context makes it unnecessary. We thus tried employing top-k and top-p sampling [Holtzman *et al.*, 2019], where unlikely tokens, according to rank or probability mass, are removed before the remaining probabilities are rescaled by softmax. With $p \in [0.9, 0.99]$ we saw fewer mistakes but also less variety. This could be countered by increasing the temperature above 1.0, but we observed the melodies were still drifting. Even a nudge to these hyper-parameters led to wildly different results and finding the right balance was difficult.

This led to our development of the beam search sampling described in Sec. 3. Beam search provided melodies with more direction and virtually no mistakes but, as noticed for NLP [Holtzman *et al.*, 2019], makes for less "surprising" outputs. Correctness and variety seem to be two different objective that are difficult to obtain at the same time. However, leaving some control over those parameter to the user can provide creative opportunities. We can provide further quality control in the output of *Tradformer* by using rejection sampling: only present a tune to a user output if it meets strict criteria. This is described more in Sec. 5.

[4]See https://github.com/mister-magpie/tradformer for the appendix and code

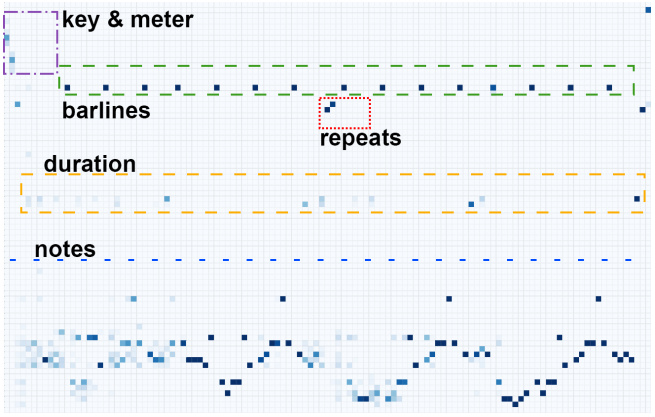

Figure 3: Portion of the parameters of the categorical distribution output by *Tradformer*. x-axis is step t. On the y-axis are tokens v_i. Darker represents more probability mass.

4.3 Parameter Analysis

We found it useful to visually analyze the parameters of the model as we developed *Tradformer*. This provided a way to see if training was proceeding correctly, and also whether the

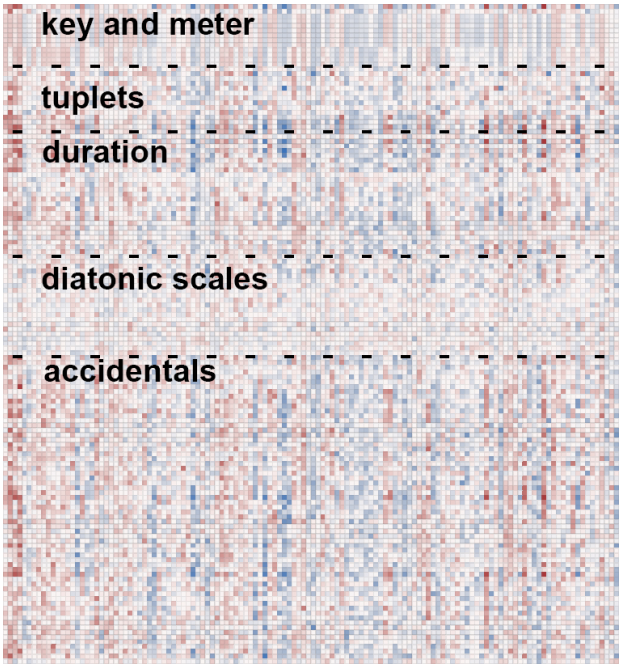

Figure 4: Weights $\mathbf{\Sigma}$ of the last layer. The dashed lines delimit tokens of the same type (e.g. key, meter, C major scale, note duration). Colors go from blue (negative) to white (zero) to red (positive).

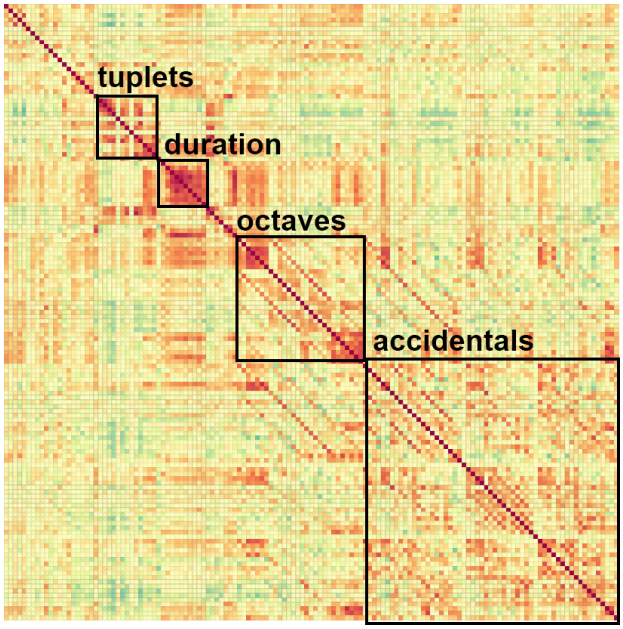

Figure 5: Cosine distances between token embeddings. Each element shows the cosine of the angle between embeddings of tokens in the vocabulary, ordered by token type. Colors cold to hot shows range [-1.0, 1.0].

model was learning relevant concepts from the music transcription sequences. Figure 3 shows an example of this with the tune from figure 2. Near the top we see high probability mass repeated in the measure line tokens. Near the bottom we see recurring patterns in pitch tokens, where confidence increases toward the end of the two phrases. The strange ending of each phrase is due to an unlikely draw from the distribution the first time, but *Tradformer* then reprises it confidently in the second part. Plotting these parameters reinforce the idea that the model has learned good representations because we see how its predictions narrow as it generates a sequence.

Figure 4 shows the matrix of the last layer $\mathbf{\Sigma}$, with rows corresponding to tokens, arranged according to token type. Each column of $\mathbf{\Sigma}$ is essentially a prototype shape of the parameter vector θ_N. *Tradformer* combines these shapes to deform the bias vector $\mathbf{b}$. We see bands in these shapes where magnitude appears to be scaled with the frequency of the token type in the training dataset. We see two stripes near the top associated with meter and key tokens. The band near the middle denotes the pitch tokens without accidentals.

Figure 5 visualizes the cosine distance between all unique pairs of token embeddings. The strong diagonal line comes from each token being colinear with itself. Several clusters and diagonal lines are clear and correspond to musical function. In particular, the off-diagonal lines relate pitches to their enharmonics or octaves. Such structures were also found in gate parameters of *folk-rnn* [Sturm, 2018].

Figure 6 shows the activity of attention heads $\hat{\mathbf{S}}_N^{(j)}$ in three layers during the generation of the tune in Fig. 2. This can reveal how the attention mechanism is referring to what it has seen to form predictions, but does not provide a complete explanation of the model [Wiegreffe and Pinter, 2019]. In

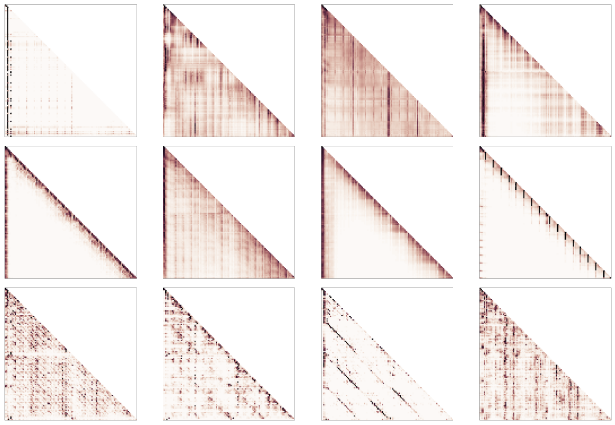

Figure 6: Attention scores $\hat{\mathbf{S}}_N^{(j)}$ from layers 1, 8 and 16 (top to bottom) for the tune in Fig. 2. Darker colors correspond to higher scores in range [0,1]

the first layer, we see one head paying constant attention to the key token (exemplified by the vertical line). In the eighth layer a head shifts its attention to the most recent measure token. At the deepest layer, we see an attention head looking at repetitions of melodic material in each section.

5 Irish to Swedish by Transfer Learning

After obtaining satisfactory results in modeling Irish music, we applied transfer learning to tune *Tradformer* to a particular style of traditional Swedish dance music: *slängpolska* – the focus of *The Ai Music Generation Challenge 2021*. Compared to the problem of modeling Irish music, this style presents a difficulty by the limited amount of data available, as well as its unique musical characteristics.

We first scraped tunes from an online resource of Scandinavian traditional music (folkwiki.se) that are tagged as "slängpolska" or had the word in its title. We then filtered incomplete and duplicate tunes, removed irrelevant material, and formed a vocabulary compatible with the Irish music dataset used to train *Tradformer*. Since slängpolskor have many semiquavers, their transcription is simplified by use of a field denoting the semiquaver as the base duration. In the Irish dataset the base duration is a quaver. We thus add a base duration token to the vocabulary. We remove vocabulary elements that are rare in the Irish dataset, and add an explicit padding token [Roccetti *et al.*, 2019]. The size of the vocabulary thus decreased from 137 to 128. To retain as much data as possible, we set the maximum sequence length to 512. The final training dataset has 593 tunes. We fine-tuned *Tradformer* for 30 epochs on the slängpolska data as more training could result in overfitting.

We see that the model seems to learn characteristic rhythmic patterns, as well as the harmonic language of slängpolskor (mostly in major and harmonic minor modes). Figure 7 shows one tune output from this model, which is considerably different from the tune shown in Fig. 2.

We observed that the outputs of the tuned model suffered from problems that can be traced back to the lack of data. Most notably, certain patterns tended to be repeated too often. The model had a tendency to get "stuck" in cycles of

two bars – a behavior not seen in either the Irish or Swedish datasets. We also found miscounted measures. The Swedish-tuned *Tradformer* would create odd-length sections, but not as often as seen in the Swedish dataset. Furthermore, we found it hard to balance between interesting output and the risk of too much repetition. Hallström et al. [2019] report that fine-tuning a *folk-rnn* model on the whole FolkWiki database (not just slängpolska) also presented difficulties in adaptation, and generated melodies described as unfocused.

The evaluation procedure in *The Ai Music Generation Challenge 2021* motivated us to employ rejection sampling to filter out generated material that does not meet strict criteria, i.e., improper meter, rhythm and poor structure. We devised a series of conditions and generated new tunes until the model produced the 1000 tunes required in the challenge submission. Inspired from the analysis by Sturm and Ben-Tal [2017], we formed criteria by looking at statistics from the original dataset and choosing reasonable values. One condition was that all bars must have the correct number of notes. Another condition was that a tune does not contain too many repeated measures, and in particular too many repeated couples – as this type of repetition happens frequently in the generated material but not in the training data. We also checked the tune length and pitch range, along with the number of bars in each section. Very short or lengthy sections show a generation that has gone amok. Rejection sampling discarded around one tenth of the outputs, evidencing the baseline quality of the model.

The five human judges of the challenge graded ten randomly selected tunes for each of six participants along four dimensions: danceability, stylistic coherence, formal coherence and playability. Output generated by *Tradformer* received in total 88 As, 61 Bs, 37 Cs, and only 3 Fs. The next closest model received 58 As, 67 Bs, 55 Cs, and only 8 Fs. The benchmark – which is just *folk-rnn* fine tuned on FolkWiki [2019] received in total 42 As, 50 Bs, 49 Cs, and 13 Fs. The appendix contains the *Tradformer* tunes and their scores. The lowest scoring *Tradformer* outputs all feature excessive repetitiveness, due to rejection sampling being sidestepped by slight variations in each "repeated" measure.

6 Discussion

The starting point for *Tradformer* was a minimal PyTorch implementation of GPT-2 [Radford *et al.*, 2018].[5] From there proceeded several iterations of design, testing and analysis. It was important early on to employ methods of evaluation by which we could gauge progress. This included human evaluation of outputs as well as visualization of the internals of the model. The inspection of loss plots gives only a limited picture of the success of the system. We found that musical quality improved significantly after carefully considering how we sample from the model. The resulting model, *Tradformer*, appears equal in quality to the LSTM *folk-rnn*, but using fewer parameters (3,205,504 compared to 5,599,881). Furthermore, the fine-tuning of *Tradformer* on a Swedish music dataset appears to have been more successful than the fine-tuning of *folk-rnn*. One limitation of *Tradformer* that *folk-rnn* does not have is that the former is limited to sequences of a maximum length. This can be addressed in future work.

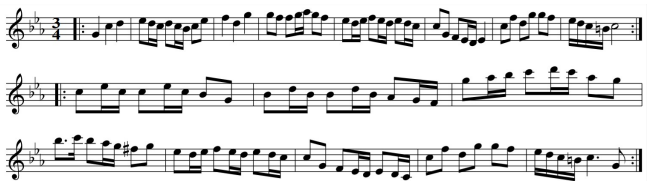

Figure 7: One output generated by the Swedish-tuned *Tradformer*. While the melodic elements are all there sometimes the model seems to drift or makes strange musical choices.

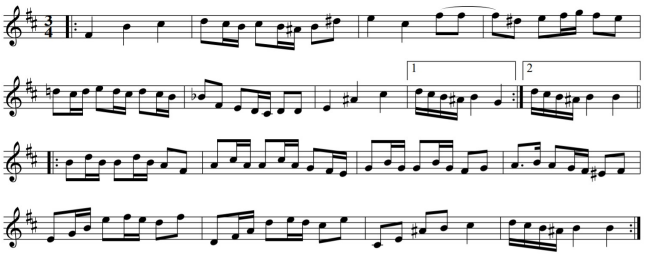

Figure 8: Modifications to the *Tradformer* output seen in Fig. 7 results in a nicer tune, which we title "Ugglas Polska".

The application of machine learning to modeling music sequences is motivated by many reasons [Pearce *et al.*, 2002], not the least of which is augmenting the ways in which music can be created. Toward this end, a measure of usefulness of such systems is how much they motivate or inhibit music creation. As an example, the tune in Fig. 7 is interesting and enjoyable, and in the neighborhood of a good tune. We titled this piece *Ugglas Polska* (Swedish for "Owl's Polka"), and performed it on our instruments, after agreeing on a number of changes to make it into a more musically appealing piece.[6] Figure 8 shows the tune after making some changes: transposing from *C minor* to *B minor* (more appropriate key for the instruments we play); introduction of some tonal ambiguity through the use of D#; bar 6 slightly altered to include an A#; bar 7 made to outline the dominant chord resolving in bar 8; bars 11, 13 and 14 replaced to develop and vary the pattern presented in bars 9 and 10; bar 12 transposed down an octave in order to avoid jumps and continue the phrasing; bars 15 and 16 rewritten to revisit the ending of the first part to create a satisfying conclusion. Such adjustments to generated material, which are not always necessary fixes but rather aesthetic decisions, can be part of the workflow of a composer.

7 Conclusion

This paper introduces *Tradformer*: a transformer-based architecture applied to modeling traditional dance music from Ireland and Sweden. Particular focus is given to the variety of engineering decisions that considerably improved performance, and provide reusable insights for modeling sequences. Namely visualization combined with human expertise prove fundamental in evaluating generative models, and provide a

[5] https://github.com/karpathy/minGPT

[6] https://bit.ly/ugglas_polska

richer view on model performance than metrics like cross-entropy. Periodically visualizing the internal structure of the model provides useful information, sometimes confirming whether the model is learning relevant concepts, and sometimes motivating the refinement of the training data and sampling strategies. Our development of *Tradformer* also highlights how fundamental the role of human expertise is when designing machine learning models. The complexity of the task and datasets, together with the lack of strong theoretical boundaries for deep neural networks (at least at the moment) make invaluable the intuition that a domain expert brings to the table. Since measuring music quality with respect to a corpus is not clear save for a comparison of basic descriptive statistics, the input from an expert can shed light on whether one is making progress.

Acknowledgments

This work was supported in part by the grant ERC-2019-COG No. 864189 MUSAiC: Music at the Frontiers of Artificial Creativity and Criticism. Thanks to Nicolas, Laura, Marco and Joris.

References

[Branwen, 2019] Gwern Branwen. Gpt-2 folk music. https://www.gwern.net/GPT-2-music, 2019.

[Casini *et al.*, 2018] Luca Casini, Gustavo Marfia, and Marco Roccetti. Some reflections on the potential and limitations of deep learning for automated music generation. In *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pages 27–31, 2018.

[Devlin *et al.*, 2018] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[Dosovitskiy *et al.*, 2020] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[Hallström *et al.*, 2019] Eric Hallström, Simon Mossmyr, Bob Sturm, Victor Vegeborn, and Jonas Wedin. From Jigs and Reels to Schottisar och Polskor: Generating Scandinavian-like Folk Music with Deep Recurrent Networks. In *The 16th Sound & Music Computing Conference, Malaga, Spain, 28-31 May 2019*, 2019.

[Holtzman *et al.*, 2019] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2019.

[Huang *et al.*, 2018] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Ian Simon, Curtis Hawthorne, Noam Shazeer, Andrew M Dai, Matthew D Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer: Generating music with long-term structure. In *International Conference on Learning Representations*, 2018.

[Payne, 2019] Christine Payne. Musenet. http://openai.com/blog/musenet, 2019.

[Pearce *et al.*, 2002] M. Pearce, D. Meredith, and G. Wiggins. Motivations and methodologies for automation of the compositional process. *Musicae Scientiae*, 6(2):119–147, 2002.

[Radford *et al.*, 2018] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. https://openai.com/blog/language-unsupervised/, 2018.

[Radford *et al.*, 2019] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[Roccetti *et al.*, 2019] Marco Roccetti, Giovanni Delnevo, Luca Casini, Nicolò Zagni, and Giuseppe Cappiello. A paradox in ml design: less data for a smarter water metering cognification experience. In *proceedings of the 5th EAI international conference on smart objects and Technologies for Social Good*, pages 201–206, 2019.

[Shaham and Levy, 2021] Uri Shaham and Omer Levy. What do you get when you cross beam search with nucleus sampling? *arXiv preprint arXiv:2107.09729*, 2021.

[Sturm and Ben-Tal, 2017] Bob L Sturm and Oded Ben-Tal. Taking the models back to music practice: Evaluating generative transcription models built using deep learning. *Journal of Creative Music Systems*, 2:32–60, 2017.

[Sturm and Maruri-Aguilar, 2021] B. L. T. Sturm and H. Maruri-Aguilar. The Ai Music Generation Challenge 2020: Double jigs in the style of O'Neill's "1001". *Journal of Creative Music Systems*, 2021.

[Sturm *et al.*, 2016] B. L. Sturm, J. F. Santos, O. Ben-Tal, and I. Korshunova. Music transcription modelling and composition using deep learning. In *Proc. Conf. Computer Simulation of Musical Creativity*, Huddersfield, UK, 2016.

[Sturm, 2018] B. L. Sturm. How stuff works: LSTM model of folk music transcriptions. In *Proc. Joint Workshop on Machine Learning for Music, ICML*, 2018.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[Wiegreffe and Pinter, 2019] Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, 2019.