Folk the Algorithms: (Mis)Applying Artificial Intelligence to Folk Music

16

Bob L. T. Sturm and Oded Ben-Tal

16.1 Introduction

This chapter motivates the application of Artificial Intelligence (AI) to modeling styles of folk music. In this context, we focus particularly on questions about the meaningful evaluation of such AI, and argue that music practitioners should be integral to the research pursuit. We ground our discussion in specific music AI that model symbolic transcriptions of traditional dance music of Ireland and Scandinavia. Finally, we discuss several ethical dimensions of such work. After reading this chapter, the reader should have a grasp of approaches to modeling music data, evaluating those approaches, and critically considering wider aspects of the application of AI to music.

Our foray into modeling and generating folk-like music began modestly as a humorous exercise one weekend in 2015 after reading Andrej Karpathy's entertaining blogpost, "The unreasonable effectiveness of recurrent neural networks" [46]. Karpathy shows how Long Short-Term Memory networks (LSTM)—a particular kind of Recurrent Neural Network (RNN)—can be trained to generate novel text one character at a time resembling Shakespeare, Wikipedia articles, and even formatted computer code. How well would such models work for music? Since Karpathy included computer code with his blogpost to reproduce his experiments, it would be

B. L. T. Sturm (✉)
Tal, Musik och Hörsel (Speech, Music and Hearing), School of Electronic Engineering and Computer Science, Royal Institute of Technology KTH, Lindstedtsvägen 24, 100 44 Stockholm, Sweden
e-mail: bobs@kth.se

O. Ben-Tal
Department of Performing Arts, Kingston University, London KT1 2EE, UK

E. R. Miranda (ed.), *Handbook of Artificial Intelligence for Music*,
https://doi.org/10.1007/978-3-030-72116-9_16

a simple matter to just replace the Shakespeare data with music—all we needed was a dataset of music expressed as text.

We downloaded an online collection of transcriptions of Irish traditional dance music from the website thesession.org [48] expressed in *ABCnotation* [1]—a text-based shorthand representation invented to help memorize folk music. We extracted each transcription to create a text file of 423,249 lines and 13,519,069 characters. Below is an extract from that file showing three settings of an Irish polka titled "The Ballydesmond":

```
T: Ballydesmond, The
M: 2/4
L: 1/8
K: Ador
|:E>A AB|cd e2|G>F GA|GF ED|
|E>A AB|cd ef|ge dB|A2 A2:|
|:a2 ab|ag ef|g2 ga|ge de|
|e<a ab|ag ef|ge dB|A2 A2:|

T: Ballydesmond, The
M: 2/4
L: 1/8
K: Ador
|:"Am"EA AB|cd e2|"G"G>F GA|GE ED||"Am"EA AB|B1/2c1/2d ef|"G"g1/2
    f1/2e"Em" dB|"Am"A2 A2:||:"Am"a>g ab|ag ef|"G"g>f ga|ge d2||"
    Am"ea ab|ag ef|"G"ge "E7"dB|"Am"A2 A2:||:"Am"c2"Em"Bc1/2B1
    /2|"Am"AB1/2A1/2G>A|"G"Bded|g2gd||"Am"e1/2g1/2a"Em"ge|"G"
    dBGA1/2B1/2|"Am"ce"Em"dB|"Am"A2 A2:||:"Am"eaag1/2e1/2|"G"
    dgge1/2d1/2|"Am"eaab|"Em"g2ed||"Am"ea"Em"g1/2a1/2g1/2e1/2|"G"
    dBGA1/2B1/2|"Am"ce"Em"dB|"Am"A2 A2:|

T: Ballydesmond, The
M: 2/4
L: 1/8
K: Ador
|: A/G/ |EA A>B | cd e2 | G/A/G/F/ G>A | GE ED |
EA- A>B | cd e>f | g/f/e dB | A2 A :|
|: B/d/ |ea a>b | a/b/a/g/ ef | g>f ga | ge ed |
ea- a>b | ag ef | ge dG | A2- A :|
```

ABCnotation is described more thoroughly in Sect. 16.3.2, but for now all one needs to understand is that training an LSTM network on this text file means adjusting its parameters such that it is likely to output a correct character given all the characters it has seen up to that point. Taking the first setting of "The Ballydesmond" as an example, this means making the LSTM network likely output '`:`' given the input '`T`'; and then output a space given '`T:`'; and then output a '`B`' given '`T: `'; and then outputting '`a`' given '`T: B`'; and so on.

Using Karpathy's code [46], we trained an LSTM network on batches of 50-character excerpts of this text file. We then had the trained model—which we call *folk-rnn (v1)*—generate tens of thousands of new transcription and published some of these online [28]. Here is one example generated by *folk-rnn (v1)*:

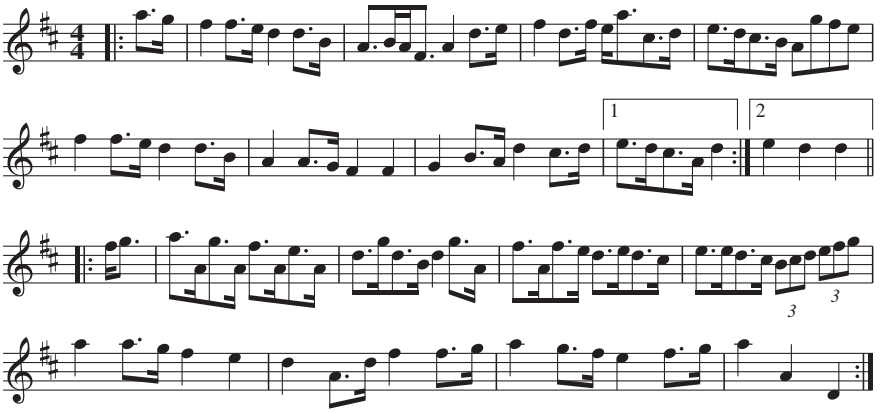

Fig. 16.1 Notation of "The Mal's Copporim" generated by *folk-rnn* (v1), which exemplifies many of the local and global conventions of the transcriptions in its training data

```
T: The Mal's Copporim
M: 4/4
K: Dmaj
|: a>g | f2 f>e d2 d>B | A>BA<F A2 d>e | f2 d>f e<ac>d | e>dc>B
    Agfe |
f2 f>e d2 d>B | A2 A>G F2 F2 | G2 B>A d2 c>d |[1 e>dc>A d2:|[2 e2
    d2 d2 ||
|: f<g | a>Ag>A f>Ae>A| d>gd>B d2 g>A| f>Af>e d>ed>c| e>ed>c (3
    Bcd (3efg |
a2 a>g f2 e2 | d2 A>d f2 f>g | a2 g>f e2 f>g | a2 A2 D2 :|
```

Figure 16.1 shows the notation of this transcription. While the melody does not sound particularly Irish, it is convincing and original, has a typical AABB structure, shows rhythmic consistency and novelty, repetition and variation, and uses cadences appropriately. The first part also has been given two endings. The model has even created a unique title: neither "Mal" nor "Copporim" appear in the training data.

We synthesized over 35,000 of these generated tunes using a variety of instruments common to Irish traditional music, and created *The Endless Traditional Music Session* website to stream these results. Figure 16.2 shows a screenshot. Every five minutes a new random set of seven tunes would appear in rotation. We posted a message about this on the discussion forum of the website from which we got the data (https://thesession.org/discussions/37800). Most replies were critical: user `hnorbeck` writes, "Interesting, but the results sound rather dull." `Ergo` writes: "I listened to a couple and they sound – weird. I mean the melodies themselves, not the production. Nothing going on. I think you'd have to get a good musician or two to actually play a few of these for them to make sense, if they can make any sense." `AB` writes, "Basically it's crude turntabling without the sense of a musician familiar with the significance of various motifs & phrases." `ceolachan` notes a disconnec-

Fig. 16.2 Screenshot of *The Endless Traditional Music Session* webpage, which served up a random set of seven tunes generated by the music AI *folk-rnn (v1)*. The titles and group names were generated by the model as well

tion between the music and its function: "Teach it to dance first?!" A few comments describe trying to play some of the generated tunes, e.g., `Mark Harmer` writes,

> I've had a romp round the archive of mp3s. It's a slightly surreal experience, like you are listening to the output of someone locked in a cell and forced to write tunes! …Interesting to listen to a few - normally you know pretty much immediately whether a tune's going to be good or not, but there is quite a lot of variation during the tune - not "totally unexpected variation" but not simple repetition either. In [The Mal's Copporim], the first two phrases are quite fun as a generative idea to "human-compose" the rest of it! I know that's not quite the point of course. Still had fun trying the opening of this one on the harp …

Regardless of the fact that many of the generated melodies did not sound like authentic Irish traditional tunes, we did not have difficulties finding examples that

were plausible and interesting enough for composition. One of the first examples is Sturm's 2015 electroacoustic composition, "Eight short outputs generated by a long short-term memory network with three fully connected hidden layers of 512 units each trained on over 23,000 ABC transcriptions of session music (Irish, English, etc.), and arranged by my own 'personal' neural network trained on who knows what for who knows how long (I can't remember any of the settings)" [68]. Our exploration eventually led to many interesting research questions that motivated more serious and deliberate work in a variety of directions, and which resulted in many conference papers [35,69,70,73], journal articles [43,71,74,75], workshops and concerts, a professionally recorded music album, "Let's Have Another Gan Ainm" [72], media attention, and significant research funding including AHRC No. AH/R004706/1 (Engaging three user communities with applications and outcomes of computational music creativity) and ERC-2019-COG No. 864189 (MUSAiC: Music at the Frontiers of Artificial Creativity and Criticism).

In this chapter, we survey several directions of our research in music AI. In the next section, we discuss how folk music provides exciting avenues for research in machine learning and AI, and survey past work in emulating folk music with computers. Section 16.3 describes several versions of *folk-rnn* that we have created, motivated by questions of music and engineering. Section 16.4 evaluates some of these systems to gauge how successful they are, and, more broadly, how useful they can be for music creation. Finally, Sect. 16.5 discusses some of the ethical dimensions of our research. We hope that this chapter provides an intriguing look at how research in music AI can be accomplished in ways that are productive and respectful of the practices from which it draws.

16.2 Music Artificial Intelligence and Its Application to Folk Music

Music AI involves engineering machines that can perform tasks that would normally require human music intelligence or ability. Examples include: recognizing musical instruments, segmenting music, recommending music, identifying musical characteristics like melody, harmony, and structure, expressively performing music, transcribing music, composing, accompanying, and improvising. The commercial and cultural applicability of such systems translates to considerable impacts, both positive and negative [43,75]. Technical details of such systems can be found in several texts; e.g., Lerch [52], Müller [57], Knees and Schedl [49], Dean and McLean [20].

Applying computational modeling to create music has a rich history beginning in the late 1950s [40]. Much of that work is centered on the emulation of heavily theorized musical styles such as the chorales of J. S. Bach; e.g., Ebcioğlu [22], Hild et al. [38], Hadjeres et al. [34]. Comparatively little work, however, has been devoted to modeling and emulating folk music. This is surprising for several reasons. For traditions that are still practiced, like Irish music, there exists a lot of data with

which music AI can be trained. Much of this music data is free of copyright as well. Even though folk music can lack explicit rules, it often still has implicit conventions that can motivate decisions for modeling and evaluation. Irish traditional music is unique in the sense that expert practitioners can be found in many places around the world. This makes it possible to involve practitioners in an evaluation process. Such research can provide starting points for exploring the emulation of other music styles, and for studying the computer augmentation of human creative practices. Sections 16.3 and 16.4 give several examples of the above; but first, we survey past research (other than our own) in the computational modeling of folk music.

16.2.1 1950s–60s

The first reference we can find applying machines to generating folk-like music is given by Hiller [39], who mentions work performed around 1951 but not published until a decade later: Olson and Belar [60] programmed a machine to generate melodies in the style of those written nearly a century earlier by American composer Stephen Foster, himself borrowing from folk songs at that time. Olson and Belar [60] describe their machine as meant to be an "aid" to the composer, "in his search for melody which is the essence of most music." This work occurred only a few years after machines started to be applied to analyzing folk melodies, as in the work of Bronson [8].

Cohen [12] mentions work from 1955 on the generation of music by a first-order Markov chain with note transition probabilities found by analyzing "Western cowboy songs". This work appears to never have been published. Pinkerton [66] takes a similar approach but with 39 nursery tunes. These works appear to be motivated by the mathematical study of music from the perspective of information theory.

Brooks et al. [9] is perhaps the most thorough early investigation of melody generation by computer models with parameters found from existing music. They analyze the melodies in 37 common-meter hymns, and build Markov chains having orders from one to eight. They also impose a variety of constraints on the generation process, such as note durations, and starting and ending pitches, and generate 600 new melodies of eight measures. They discuss some of the results in terms of pitch range, melodic contour, intervalic content and singability, and the reproduction of melodies in the training material. Similar to Pinkerton [66], Brooks et al. [9] explore the use of music synthesis to test the completeness of their statistical analysis of music.

Hiller [39], in a section titled, "Algorithms for generating folk tunes", observes that much music generation synthesis work up to that time had occurred in the Soviet Union, Hungary and Czechoslovakia, and gives several examples. For instance, Havass [36] analyze 100 folk songs collected by Hungarian composer Zoltán Kodály, and synthesize new melodies using a Markov model built from that analysis. They present no musical examples in the text, but propose to play five generated melodies from magnetic tape at the August 1964 conference of the "International Folk Music Council" (where Zoltán Kodály delivered the keynote address). It is unclear if this materialized since a report about the conference makes no mention of this in the

schedule. The Hungarian Academy of Sciences, the institute under which Havass worked, is also noted to be studying folk dancing from computational perspectives [27].

16.2.2 1970s–90s

Lieberman [53] proposes testing whether a given set of statistics is sufficient for describing a melodic style by generating melodies using those statistics and comparing them with real ones. They briefly discuss applying such an approach using Markov models with parameters derived from analyses of Javanese melodies, and motivate the search for statistics that are more descriptive of that style of music since the results are poor. A similar argument of studying the completeness of a set of musical rules is found in Sundberg and Lindblom [76], who study the generation of melodies according to grammatical rules found from analyzing Swedish nursery songs by a specific composer, as well as a small set of folk tunes.

Cope [19] applies his *Experiments in Music Intelligence*—an approach he developed for imitating Western composers such as Bach or Chopin by focusing on patterns and their variation—to gamelan gong kebyar, based on transcriptions into Western notation. These, as he observes, abstract away the tuning system as well as the timbral qualities—both rather important attributes in this music. According to Cope, the generated outputs were considered acceptable to gamelan musicians. He acknowledges that this endeavor may be biased because of the grounding in Western musical concepts that only partially capture the gong kebyar music.

Mozer [56] proposes music modeling and generation using artificial neural networks and a musically-informed representation. He presents results based on artificial melodies, melodies by Bach, and 25 "traditional European folk melodies" from a 17th century collection of melodies for recorder. He provides an example of a melody generated in the latter style, but performs no deeper evaluation.

16.2.3 2000s–10s

Eck and Schmidhuber [24,25] investigate the success of a long short-term memory network (LSTM) [42] in modeling and generating "twelve-bar blues"—a melodic form following a specific harmonic progression. The training data was constructed by concatenating together 12 common-meter measures of crotchets from a scale of 6 pitches each within a particular harmonic context. Each measure was composed by the authors. The results are evaluated by informal listening, and by comparison with melodies generated by random walks.

Lapalme [51] appears to be the first to train music AI on traditional music from Ireland and England, which is extended in Eck and Lapalme [23]. They created one dataset from 56 Irish tunes downloaded from thesession.org [48], and a second dataset of 435 tunes from the so-called *Nottingham* dataset, which is a collection of about 1035 British folk music pieces [29]. All transcriptions in this dataset are notated in

common meter. The authors quantize each melody as a sequence of quavers, and build a representation that links notes to others in the same metric position occurring previously. They train an LSTM network to predict the next pitch of a sequence given the previous pitch, and the pitches occurring at the same beat position in the three preceding measures. They test the accuracy of each model in continuing the first eight measures from tunes in a validation partition of their datasets.

Spiliopoulou and Storkey [67] also use the *Nottingham* dataset and explicitly state that their motivations are to study the success of machine learning systems in capturing and imitating the structures in these melodies, as well as analyzing what musical concepts the models learn and how that knowledge is represented. They compare three machine learning models trained on 117 melodies. As done in Eck and Lapalme [23], they encode a melody as a sequence of events quantized with a quaver time step. The events are either pitch (one of 24), silence, or a continuation. The models are built to predict events occurring several time steps ahead of conditioning events. Their analysis of some of the components of the trained models show them to have acquired sensitivity to meaningful musical features, e.g., triads, arpeggiation, and scalar movement.

The *Nottingham* dataset appears in several other published works in machine learning; e.g., Paiement [61], Boulanger-Lewandowski et al. [7], Bengio et al. [5], Pascanu et al. [62], Goel et al. [30], Chung et al. [11], Yu et al. [78], Johnson [44], Bacciu et al. [3]. The only form of evaluation of the resulting models appearing in these works involves computing how well-trained models predict real sequences held out from training. These works contain no discussion of the usefulness of such systems for music creation.

A unique project involving AI and folk music is LIVINGSTON [54]: "an artificially intelligent, digital organism capable of accessing the totality of the history of Canadian folk music (among other corpuses) and generating new yet hyper-authentic Canadian folk objects via her/his algorithmic agents and compression formats." This system seems to only generate lyrics and chord progressions, and the two volumes of recorded music produced with it—titled "Artificially Intelligent Folk Songs Of Canada"—is performed by humans. Not much more can be surmised from existing resources at this time.

Herremans et al. [37] train Markov models of different orders on transcriptions of 37 melodies performed on the traditional Ethiopian lyre (called a *bagana*). This instrument has 10 strings, only 6 of which are sounded. Each played string is associated with a different finger: five on the left hand, and the index finger of the right hand. A melody can then be represented by the finger that plucks the string. The authors explore a variety of metrics to gauge the fitness of the resulting models. This work is the first we can find in which melodies generated by models are qualitatively assessed by a practitioner of the music style used for training.

Colombo et al. [14] train music AI on 2,158 Irish traditional melodies transcribed by Norbeck [58]. They represent a melody as a sequence of elements: paired pitch and duration values, as well as "ending" and "silence". They normalize the melodies by transposing them to be in C major or A minor, and scaling all durations based on the frequency of the most common duration. They propose to model a melody by two

recurrent neural networks, one modeling a conditional distribution on the durations, and another modeling a conditional distribution on the pitches and duration—that is, one network predicts the next duration based on the previous durations generated, and the other network predicts the next pitch based on the previous pitches generated and the duration of the pitch to be generated. They evaluate the resulting model by observing how it continues a given seed, either 2 notes or 8 full measures. The melodies shown in the paper are aimless and bear little resemblance to Irish traditional music.

Colombo et al. [15] extend the approach taken in Colombo et al. [14]. They again propose using two recurrent neural networks, but this time one models the conditional distribution of durations given all previous durations and the current pitch; and the other models the conditional distribution of pitches given all previous pitches and the next duration. They create a dataset combining 2,160 Irish melodies from Norbeck [58], and 600 Klezmer melodies from Chambers [10]. In this case, they do not transpose all melodies to a common key. They propose a measure of tune novelty with respect to a collection based on the co-occurrence of subsequences in each. The article is accompanied by synthesized examples, using harp sound for the Irish ones and clarinet for the Klezmer, thus accentuating the differences. Several examples have aimless melodies that do not sound Irish, and some of the Klezmer examples veer off course.

Colombo et al. [16] propose a different music representation from their past work. Each note in a sequence is given by a tuple: pitch, duration, and time offset relative to last note. They propose modeling the joint probability of a sequence of notes as a product of three conditional distributions. Each of these distributions is modeled as a layer in a recurrent neural network (RNN), with conditioning supplied after sampling from the output of each of the three hidden layers. In the first step, their model samples a time offset; then the model samples a duration; and finally, the model samples a pitch. They train models on a variety of datasets, including *Nottingham*. For their trained model, they measure the mean likelihood of melodies of a validation dataset. They also link to a website where one can listen to dozens of sound files created from synthesizing the generated music.

Goienetxea and Conklin [31] are motivated by the challenging problem of creating a music AI that can compose melodies with "coherence", or sensible long term structure coming from the creative development of basic material. They focus on modeling the structures found in a set of 2,379 Basque folk melodies [21]. Their music representation uses what is called "multiple viewpoints" perspectives, a description of music at several levels of detail [18]. They use the resulting model to generate melodies in the same style, and describe a concert in which the audience was tasked with trying to identify which of three melodies was not computer generated.

Pati et al. [64] propose a music AI that generates material linking a given beginning and ending. They approach this by building a probabilistic model that interpolates between representations of these contexts in a latent space. They write of using a subset of size about 21,000 melodies notated with a common meter from the collection of Irish traditional music transcriptions used in Sturm et al. [73]. Since that dataset only has at most 12,593 melodies that fit this description, it is possible

the authors split up melodies into eight-measure sections. The resulting models are evaluated quantitatively in terms of model fit, and qualitatively, using a subjective listening test involving rank which of two completed melodies is preferred.

16.3 Modeling Folk Music Transcriptions with Long Short-Term Memory Networks

Building *folk-rnn (v1)* and experimenting with it motivated several interesting research questions. What would happen if we trained the same kind of model but using transcriptions expressed with a more efficient and musically meaningful vocabulary? How can we meaningfully evaluate these systems with music practitioners, both inside and outside the traditions from which the data comes? How can we measure the "musical intelligence" of these systems? How can we adapt their knowledge to other music traditions? How could such models contribute to and detract from music creation? What does the training, evaluation, existence, and use of such models mean for traditional music? How might they impact traditional music in positive and negative ways?

We have so far built several versions of *folk-rnn*. While each version is a standard LSTM network, they differ in terms of training data and music representation. In this section, we discuss the technical details of LSTM networks. We then describe several different versions of *folk-rnn*, and present some of their outputs. Section 16.4 discusses in more depth methods we have used to evaluate these models.

16.3.1 Long Short-Term Memory Networks

LSTM networks are a type of recurrent neural network (RNN) with special mechanisms to control the flow of information through it as it models a sequence [42]. It is essentially a dynamic model of a probability distribution describing what is likely to come next in a sequence it is observing. To be more explicit, say the LSTM network has observed the sequence of vectors $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t)$. It computes the *posterior probability distribution* of the next vector, $P(\mathbf{x}_{t+1}|\mathbf{x}_t, \ldots, \mathbf{x}_1)$—that is, the probability of observing $\mathbf{x}_{t+1}$ given the t observations up to that step.

Figure 16.3 diagrams an LSTM network having a single hidden layer. There can be any number of hidden layers, however. The hidden layer we use for *folk-rnn* models processes the input at time step t according to the following algorithm [32]:

$$\mathbf{i}_t \leftarrow \sigma\left(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{b}_i\right) \tag{16.1}$$

$$\mathbf{f}_t \leftarrow \sigma\left(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{b}_f\right) \tag{16.2}$$

$$\mathbf{o}_t \leftarrow \sigma\left(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{b}_o\right) \tag{16.3}$$

$$\mathbf{c}'_t \leftarrow \tanh\left(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c\right) \tag{16.4}$$

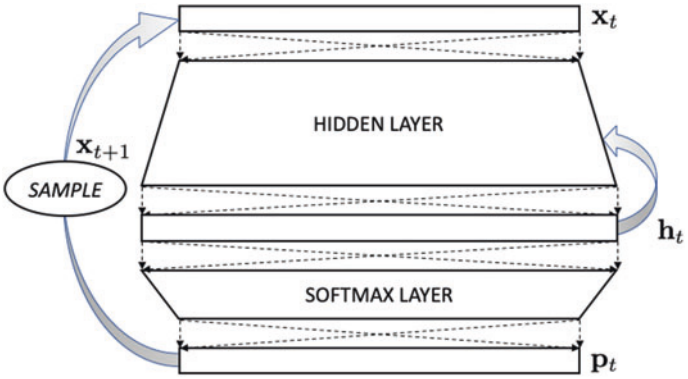

Fig. 16.3 An LSTM network with one hidden layer. An input vector $\mathbf{x}_t$ at step t is processed by the hidden layer, to form the hidden state vector $\mathbf{h}_t$ in a possibly higher dimension. This is then projected by a softmax layer to a vector $\mathbf{p}_t$, which defines the probability distribution $P(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{x}_{t-1}, \ldots, \mathbf{x}_1)$. Sampling from this distribution produces the output for the next time step, $\mathbf{x}_{t+1}$, which becomes the next input to the model when the LSTM network is generating a sequence

where σ denotes the sigmoid function

$$\sigma(x) := \frac{1}{1 + e^{-x}}$$

which is applied to each element of the vector. The hyperbolic tangent is similarly applied to each element of the vector. The vectors $\mathbf{i}_t$, $\mathbf{f}_t$ and $\mathbf{o}_t$ are called the "in gate," "forget gate," and "out gate", respectively. These encode the new information passed into the LSTM by $\mathbf{x}_t$ with the context of past information represented by $\mathbf{h}_{t-1}$. The matrices $\mathbf{W}_{x*}$ and $\mathbf{W}_{h*}$, and bias vectors $\mathbf{b}_*$, define how this information is encoded in the hidden layer. These vectors are then combined to update the "cell state" and "hidden state" of the hidden layer, respectively:

$$\mathbf{c}_t \leftarrow \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{c}'_t \tag{16.5}$$

$$\mathbf{h}_t \leftarrow \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \tag{16.6}$$

where $\odot$ denotes element-wise multiplication. This shows how updating the cell state involves modulating the cell state of the prior step with the forget gate while adding new information from the in gate. The new hidden state is a product of the out gate with a compression of the updated cell state.

The softmax layer transforms $\mathbf{h}_t$ as follows:

$$\mathbf{p}_t \leftarrow \text{softmax}\left(T_s^{-1}[\mathbf{W}_s\mathbf{h}_t + \mathbf{b}_s]\right) \tag{16.7}$$

where T_s is a user-specified parameter called *temperature*, and the softmax function is defined

$$\text{softmax}(\mathbf{y}) := \frac{\exp(\mathbf{y})}{\sum \exp(\mathbf{y})}$$

which scales the vector $\mathbf{y}$ such that its elements sum to one. The vector $\mathbf{p}_t$ specifies the posterior probability distribution $P(\mathbf{x}_{t+1}|\mathbf{x}_t, \ldots, \mathbf{x}_1)$. Sampling from this distribution produces a prediction of $\mathbf{x}_{t+1}$. If the LSTM network is generating a sequence, one need only make $\mathbf{x}_{t+1}$ the input for updating the posterior distribution, and then predict the next element of the sequence. This procedure cycles indefinitely until a stopping criterion is reached. If the LSTM network has several hidden layers, then each hidden layer after the first transforms the hidden state vector of the preceding layer according to the equations above (but with different parameters). Other architectures are possible too; e.g., where each hidden layer has access to the states of all other hidden layers [32].

The parameters of an LSTM network—the initial conditions $\mathbf{h}_0$ and $\mathbf{c}_0$ of each layer, the matrices and biases transforming the input of each layer, and the matrix and bias of the softmax layer—come from training the LSTM network to minimize a specified loss function. The loss used for *folk-rnn* models is called the *mean cross-entropy loss*. Consider a sequence s of M indices into a discrete vocabulary; e.g., 256 alpha numeric characters. Let us encode this sequence as a series of vectors, $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M)$, each dimension being zero except for the one that corresponds to the vocabulary element, which is called *one-hot encoding*. Each dimension of $\mathbf{x}_m$, and likewise the LSTM network output $\mathbf{p}_m$, refers to a particular element of the vocabulary. The goal of the network in step m of modeling sequence s using cross-entropy loss is to predict which dimension of $\mathbf{p}_m$ should be set to one. This means that at step m we want to make the network produce a vector $\mathbf{p}_m$ that looks like $\mathbf{x}_{m+1}$. In order to do that, we want the output of the network to minimize the mean cross-entropy loss over a sequence:

$$L(s) := -\frac{1}{M} \sum_{m=1}^{M-1} \log[\mathbf{p}_m]_{s(m+1)} \tag{16.8}$$

where $s(m)$ is the mth element of the sequence, and $[\mathbf{p}_m]_i$ is the ith element of the vector. Each individual term in the sum above is the cross-entropy loss at that step in the sequence. If the LSTM network produces $\mathbf{p}_m = \mathbf{x}_{m+1}$ for all elements of the sequence, then $L(s) = 0$, the smallest it can be. However, if the network produces a $\mathbf{p}_m$ which is close to zero in dimension $s(m+1)$, then $L(s)$ will become very large. Training the network with this loss entails making it move as much probability mass into the correct dimensions of the posterior distribution so as to make $L(s)$ small for most training sequences. This is accomplished by using back-propagation through time with stochastic gradient descent, and other computational techniques intended to avoid overfitting. More details are provided in Sturm et al. [73].

16.3.2 folk-rnn (v2)

The second version of *folk-rnn* [50], applies the same LSTM network architecture to the same training data as the first version [48], but uses a modified music representation. Since *folk-rnn (v1)* is trained to model blocks of text one character after another, it has to learn that some characters can have different functions. For instance, 'E'

can refer to a letter in a title, a pitch, part of a pitch (e.g., '_E'), or part of a key (e.g., 'K:Emin'). This ambiguity means *folk-rnn (v1)* has to learn the different contexts in which each character can appear. Furthermore, modeling blocks of text in a document is not necessarily modeling music transcriptions. For instance, some training blocks could begin in the middle of a transcription. *folk-rnn (v1)* also had to learn about the many keys in which the training tunes are transcribed. Most Irish traditional music uses four modes: ionian (major), aeolian (natural minor), dorian, and mixolydian; but these can involve many keys; e.g., G, D, and A major, E and B minor, D and A mixolydian, A and E dorian. To create *folk-rnn (v2)*, we thus set out to train an LSTM on music transcriptions expressed by a vocabulary where each of its symbols has only one meaning.

Before we discuss the vocabulary we designed, we review the representation used in the data collected from thesession.org [48]. ABCnotation [1] was designed to compactly describe the "bones" of a folk tune. It is important to know that in Irish traditional music, a transcription of a tune only provides a basic structure. Rarely is a melody played as notated; performers elaborate upon the "bones" using ornamentation, variation, harmony, and rhythmic push and pull to give "lift" [26]. In ABCnotation [1], information fields are marked with a capital letter followed by a colon; e.g., 'T:' provides the title; 'M:' specifies the meter; 'L:' specifies the base duration of a note without an explicit duration marker; 'K:' specifies the key. Following these fields is the tune body, which notates the melody. Pitches within the given key are specified by a letter, which may be sharped or flatted by preceding it by '^' or '_', respectively. In the key of C major, 'C' is middle C, while 'C,' is an octave below, 'c' is an octave above, and 'c'' is two octaves above. More commas and single quotes can be added to lower or raise the pitch. Harmonic accompaniment is specified in double quotes, such as "Am". Multiple pitches sounding at the same time are grouped between square brackets, such as '[Gd]'. When note durations are specified explicitly, they are either numbers after the pitch (e.g., '2'), or symbols: '/' is shorthand for '1/2', while 'A > B' steals time from the second pitch and gives it to the first, conversely 'A < B' does the opposite, and '(3 EFG' indicates a triplet. Otherwise, note durations take on the value specified by the 'L:' field. Finally, the symbol '|' shows a measure line, '|:' and ':|' are beginning and ending repeat signs, and '|1' and '|2' are first or second endings, respectively. Many other symbols are possible.

To address the issue of ambiguity in ABC representations, we designed a vocabulary of musical *tokens*, where each token represents only one thing. The vocabulary we designed consists of 137 tokens grouped into seven types (examples given in parentheses): *meter* ('M:6/8'), *key* ('K:Cmaj'), *measure* (':|' and '|1'), *pitch* ('C' and '^c'), *grouping* ('(3'), *duration* ('2' and '/2'), and *transcription* ('<s>' and '<\s>'). We transposed all transcriptions to have a root note of C as well, so that a model would only need to learn about the four typical modes. We also removed titles, harmonic specifications, grace notes, ties and slurs, and other markings. As an example, *The Ballydesmond Polka* given in the Introduction becomes the following sequence of 90 tokens in the new representation:

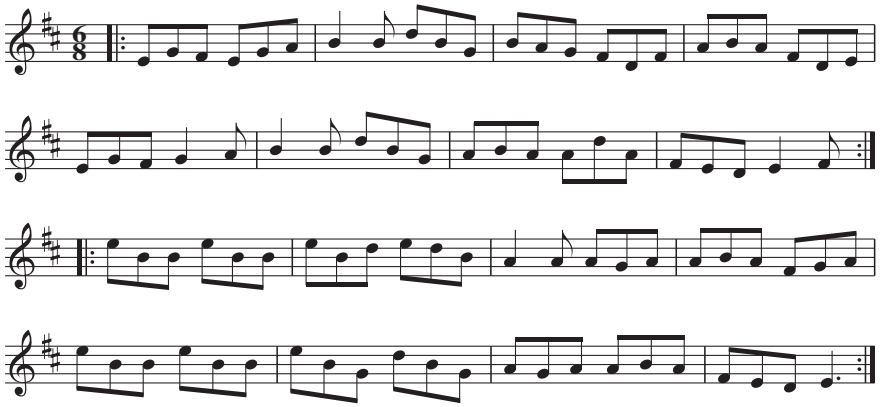

Fig. 16.4 Notation of transcription #18727 generated by *folk-rnn* (v2), which can be found in "The folk-rnn (v2) Session Book Volume 7 of 10" [28]. We transpose it here to E dorian from C dorian

```
<s> M:2/4 K:Cdor |: G > c c d | e f g 2 | B > A B c | B A G F | G
     > c c d | e f g a | b g f d | c 2 c 2 :| |: c' 2 c' d' | c'
    b g a | b 2 b c' | b g f g | g < c' c' d' | c' b g a | b g f
    d | c 2 c 2 :| </s>
```

Each token is demarcated by a space. The tokens '<s>' and '</s>' signify the beginning and ending of a transcription, respectively.

In addition to transposing and tokenizing the collection of transcriptions we retrieved from thesession.org [48], we performed a significant amount of cleaning: removing comments masquerading as tunes, removing jokes (e.g., Cage's "4m33s"), removing chord progressions, and fixing as many human counting errors as possible. We removed all transcriptions that had explicit changes in meter or key so that all transcriptions followed the same pattern: meter, mode, and tune. The encoded and cleaned dataset consists of a total of 23,635 transcriptions, with a total of 4,056,459 tokens, of which 2,816,498 are of the type pitch, 602,673 are of the type duration, and 520,290 are of the type measure [50].

The network architecture of *folk-rnn (v2)* is essentially the same as for the first version (having three hidden layers of 512 units each), but with input and output dimension 137. The total number of parameters in *v2* is 5,599,881. Training proceeds in nearly the same way as for the first version, but uses minibatches of 64 entire transcription sequences rather than continuous chunks of text. The *v2* model results from 100 epochs of training, one epoch being exposure to all transcriptions in a training partition. More details are provided in Sturm et al. [73].

As for the first version, we had *folk-rnn (v2)* generate tens of thousands of transcriptions and published 10 volumes of these [28]. The model is initialized with the one-hot vector representing the token '<s>', and terminates token generation when it produces '</s>'. One example output of this model is shown notated in Fig. 16.4. This transcription shows the conventional structure, rhythmic consistency, repetition and variation, and appropriate cadences. The second part goes higher in

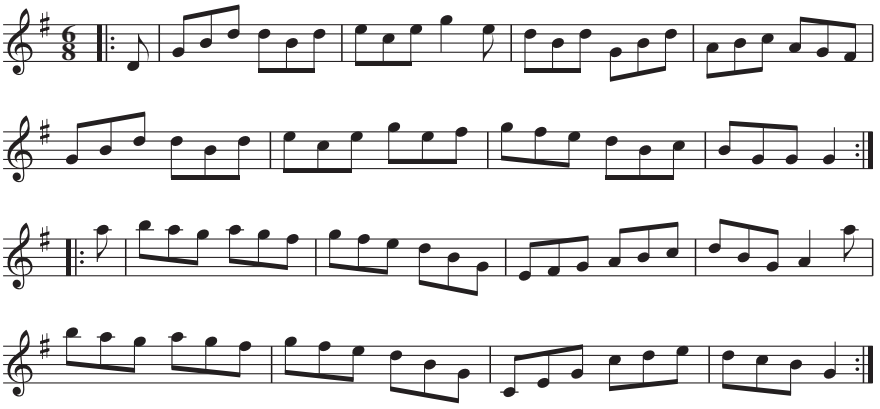

Fig. 16.5 Notation of transcription #5712 generated by *folk-rnn* (v3), which can be found in "The folk-rnn (v3) Session Book Volume 3 of 4" [28]. We transpose it here to G major from C major

pitch than the first, which is a typical characteristic of this kind of music. The two sections of the tune are also linked together well: the fourth measure of each part is similar, and the endings of both parts are the same. It also sounds like Irish traditional dance music, and is very playable on traditional instruments—providing opportunities for ornamentation and variation. Several more examples generated by *v2* are discussed in Sturm et al. [73], Sturm and Ben-Tal [71], including using the model to "autocomplete" melodic ideas.

16.3.3 folk-rnn (v3)

Although the vocabulary we designed for *v2* addresses ambiguity in ABCnotation [1], it still has redundancy. For instance, for a transcription in the mode of C major, the token '^B,' refers to the same pitch as 'C' and '=C'. In the C minor mode, the token 'E' refers to the pitch E flat above middle C, which is the same as the token '_E'. We thus decided to train an LSTM network on the same collection of transcriptions but with all pitches made explicit, and using only naturals and sharps. Furthermore, so that the model could learn about all possible pitches in the vocabulary, we added all transcriptions transposed up a half step (having a root of C-sharp). We keep the four mode tokens, but do not specify the root. This resulted in a vocabulary of size 104 tokens in the same seven types as for *v2*. In this representation *The Ballydesmond Polka* given in the Introduction becomes (with a root of C):

```
<s> M:2/4 K:Cdor |: G > c c d | e f g 2 | B > A B c | B A G F | G > c c d
    | e f g a | b g f d | c 2 c 2 :| |: c' 2 c' d' | c' b g a | b 2 b c' |
     b g f g | g < c' c' d' | c' b g a | b g f d | c 2 c 2 :| </s>
```

As for *v1* and *v2*, we had the trained *folk-rnn (v3)* generate 10,000 transcriptions, available in four volumes [28]. Figure 16.5 shows a particularly good output of this model displaying many of the conventions of the style.

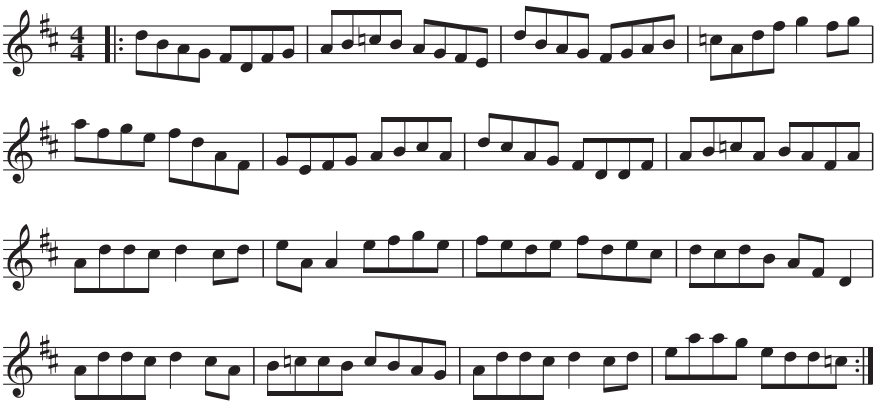

Fig. 16.6 Notation of a transcription generated by *folk-rnn (v2)* using beam search with $n = 4$ tokens selected in each step. We transpose it here to D major from C major

16.3.4 folk-rnn (vBeamSearch)

One step of an LSTM network results in an estimation of $P(\mathbf{x}_{t+1}|\mathbf{x}_t, \ldots, \mathbf{x}_1)$. However, this can be generalized to estimating a joint probability distribution of several tokens at once; e.g., $P(\mathbf{x}_{t+1}, \mathbf{x}_{t+2}|\mathbf{x}_t, \ldots, \mathbf{x}_1) = P(\mathbf{x}_{t+1}|\mathbf{x}_t, \ldots, \mathbf{x}_1)P(\mathbf{x}_{t+2}|\mathbf{x}_{t+1}, \mathbf{x}_t, \ldots, \mathbf{x}_1)$. This means that the model can be used to predict several tokens at each step by first computing the probability distribution of each token conditioned on all others, then multiplying these to form the joint conditional distribution, and finally sampling from this. As the number of tokens to be predicted simultaneously becomes large the computational complexity grows, but a strategic approach called "beam search" can make it efficient. Figure 16.6 shows a transcription generated four tokens at a time using beam search with *folk-rnn (v2)*. Henrik Norbeck, an expert in Irish traditional music and creator of a large dataset of transcriptions [58], says of this output:

> This tune sounds like it could have been composed by Paddy Fahy or Sean Ryan. There are already two tunes by them that are similar to each other — so much that in my mind they are connected — and this generated one becomes a third tune in the same class, but still a distinct tune.

16.3.5 folk-rnn (vScandinavian)

While the collection of thesession.org [48] is focused on Irish traditional music, the website folkwiki.se focuses on Scandinavian traditional music, and contains many thousand transcriptions in ABCnotation [1]. Hallström et al. [35] describes training LSTM networks using this collection of data. In this case, the amount of data acquired from folkwiki.se is an order of magnitude smaller than that used to train the

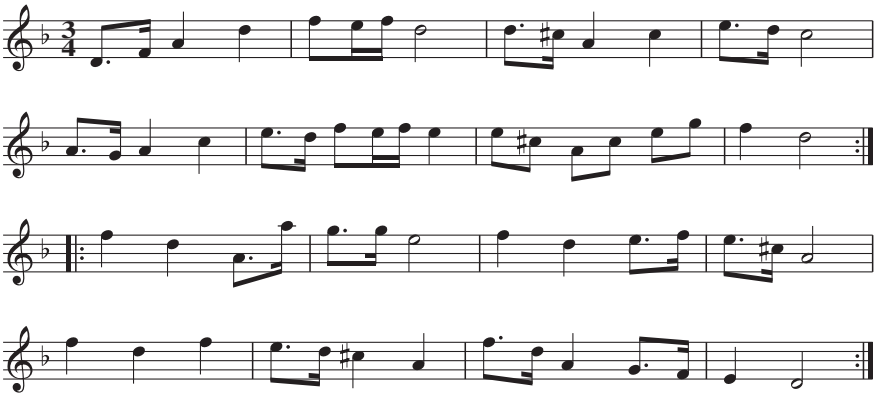

Fig. 16.7 Notation of a tune generated by *folk-rnn (vScandinavian)*, which can be found at https://themachinefolksession.org/tune/551

"Irish" versions of *folk-rnn* (4,083 transcriptions vs. 23,635). Even after designing an efficient vocabulary, models trained on only the Scandinavian transcriptions did not produce plausible melodies. To overcome this, the model was first trained on a dataset combining all transcriptions of the Scandinavian and Irish datasets. Then the pre-trained model was "fine-tuned" on just the Scandinavian transcriptions. The purpose of pretraining is to help a model learn about the vocabulary, and the syntax of the dataset. Fine-tuning then aims to adjust the model parameters to specifics of a subset. To accommodate the different ABC notation conventions in the Scandinavian transcriptions, other tokens had to be included in the vocabulary. Furthermore, the Irish transcriptions were not transposed to a common root before they were tokenized because the use of keys in the Scandinavian data follows slightly different conventions, like key changes between sections. The resulting vocabulary size of the model is 226. Figure 16.7 shows a particularly good transcription generated by the trained model.

16.4 Evaluation

One of the major questions underlying our research with *folk-rnn* is how to meaningfully analyze and evaluate such models, as well as their involvement and impact in music practice [74]. A common approach to evaluating music AI is what is often termed a "musical Turing test": listeners are presented with some music and are asked whether it came from a human or a machine. One example of this approach is by Cope [19], who asked an audience to decide whether a human-performed piece of music is by Mozart or generated by his system in the style of Mozart. More recently, Collins and Laney [13] ask listeners to compare two pieces and to identify which was written by a real composer (in this case, Bach or Chopin). Ariza [2] argues

how this terminology—"musical Turing test"—is inaccurate since the Turing test is focused on having an interactive dialogue in natural language. In contrast, the music discrimination task is very different from how we normally engage with music. Ariza [2] instead uses the terminology, "Musical Output Toy Test." We should note that in addition to the methodological problems with this approach, it also inspires the narrative pitting machines against humans, portraying AI as a threat.

Pease and Colton [65] provide an in-depth discussion of the problems with these discrimination tests in the context of computational creativity, and review alternative approaches. They first distinguish between judging the value of a generated output and evaluating the creativity of the system. They advocate focusing more on the latter in order to provide measures that can drive research forward and that are also theoretically sound. They summarise two earlier approaches, called the FACE and the IDEA models [17]. The first aims to capture aspects of the creative process: Framing information about the work, Aesthetic measures, developing Concepts, and Expressing such a concept. The IDEA model brings the viewer/listener into the equation. They propose to evaluate the effect of the experience on audience well-being (positive or negative), and the cognitive effort required to engage with the work.

Another common approach is to ask listeners to rank music generated by an AI, such as how pleasant a melody is. Problems with this include the lack of definition, and subjectivity and bias in listening. A stark example of the latter is exemplified by an unintentional experiment. An article appearing in *The Daily Mail* [33] about our work included a 30-s music excerpt from a set performed by traditional musicians at one of our workshops. Readers of the article were able to comment for a few weeks: "[The excerpt] sounds very neat. It's missing the 'human' element." "Total Crap! A foot tapping tune in 6/8 does not make it Irish. Also it feels pretty bland." "Totally lifeless without warmth." "Sounds like a robotic Irish jig…." The music excerpt posted by the Daily Mail, however, was not of a computer-generated tune, but a real traditional tune. This unintentional experiment nicely illustrates how a listener's experience of music is not just about the acoustic waves hitting their ears. Music happens at the intersection of incoming (or sometimes imagined) sounds, perception, memory, preconceptions, past experiences, social and physical environment, and myriad other factors.

Within the domain of computational creativity, Jordanous [45] proposes to capture the meaning of creativity through an analysis of existing discussion about it. She identifies fourteen components of creativity including familiar ones such as competence, originality, and value, but also aspects that are not often included in proposed definitions, such as social interactions, perseverance, and spontaneity. She suggests that evaluations should start from identifying what aspect of creativity will be assessed. The suggestion is that this would not only enable more meaningful comparisons but will also guide the choice of evaluation that matches specific components under investigation.

Yang and Lerch [77] propose the use of note-based statistical measures as a basic form of evaluation. For collections of musical works, they calculate pitch and duration ranges, note transitions histograms, and other fairly general statistics. They note that

these only apply to monophonic data, though some of the properties can be extended. The internal variability of these statistics can provide an informative profile of a dataset, either real or generated. Comparing datasets in this way can, at least, identify problems with modeling procedures, which can assist engineering. If, for example, generated melodies display markedly different statistical properties from those in the training data, this can point to specific problems with the model. Using these general measures to compare outputs of two different models can suggest the dimensions that each is successful in modeling.

Sturm and Ben-Tal [71] demonstrate five different approaches to evaluate the *folk-rnn (v2)* model: (1) comparing the statistics of real and generated transcription data ("first-order sanity check"); (2) performing basic music analysis of generated transcriptions; (3) probing a model's musical knowledge with "nefarious" initializations; (4) involving a model in music composition; and (5) performing expert elicitation with real-world music practitioners. Sturm [69,70] take another approach by attempting to reverse engineer the parameters of *folk-rnn (v2)* to understand their musical significance. Sturm et al. [74] analyze different music AI from several perspectives to determine how such models can impact music creation, and how the use of such models for music creation can inform the engineering pursuit. In the following, we look at several of these evaluation approaches.

16.4.1 Evaluation by Parameter Analysis

Sturm [70] analyzes the parameters of the input layer of *folk-rnn (v2)*, and Sturm [69] analyzes those of its softmax layer, in terms of the model vocabulary. Much more work has yet to be done to fully understand the model, but it is clear from these analyses that the model has learned some musically meaningful characteristics from looking only at data; e.g., placement of measure lines, enharmonic relationships, cadences. In a similar direction, Karpathy et al. [47] analyze the internal dynamics of recurrent models of characters in English texts, and find some parts of the models are activated near the conclusion of a sentence, quotation, or paragraph. In the case of a character model, it is difficult to draw concrete conclusions about how it is treating the elements of the vocabulary because of the ambiguity of the representation. The vocabulary of *folk-rnn (v2)*, however, is much less ambiguous by design, and so the analysis of the model becomes easier.

A unique way to analyze *folk-rnn (v2)* is by looking at how it stores and processes information in vector spaces. Figure 16.8 diagrams the procedure by which this model transforms its input into an output. Since the size of its vocabulary is 137, its input and output are vectors in $\mathbb{R}^{137}$. However, they are more restricted than that. First, since the LSTM has been trained on one-hot encoded input vectors, then the input is just one of the 137 standard basis vectors of $\mathbb{R}^{137}$. (The input can of course by any point in $\mathbb{R}^{137}$, but the model has only "seen" the 137 standard basis vectors of $\mathbb{R}^{137}$.) Second, since the output is computed by a softmax (16.7), then all elements of the output vector will be positive, and the sum of the magnitudes of the vector will be one. Hence, the output vector is a point on the positive face of the ℓ_1 unit-ball in

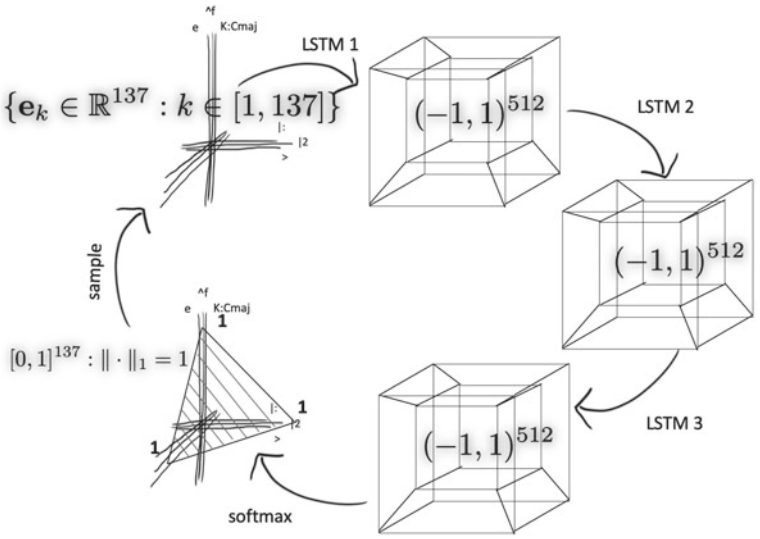

Fig. 16.8 Diagram of how *folk-rnn (v2)* is transforming information between different vector spaces. Elements of the standard basis of $\mathbb{R}^{137}$ are transformed by the first LSTM hidden layer to points in a hypercube $(-1, 1)^{512}$. The second and third LSTM hidden layers transform points in $(-1, 1)^{512}$ to points in hypercubes of the same dimension. The softmax layer then transforms points in $(-1, 1)^{512}$ to the points on the positive face of the ℓ_1 unit-ball in $\mathbb{R}^{137}$. A sampling operation then projects that point to an element of the standard basis of $\mathbb{R}^{137}$

$\mathbb{R}^{137}$. Furthermore, the ordering of the dimensions at the input and the output relative to the vocabulary is the same; i.e., the token represented by the mth dimension of the input is also represented by the mth dimension of the output.

Now let us look at the steps involved in this transformation. The first hidden layer transforms a vector of dimension 137 to a 512-dimensional vector. This is performed by the algorithm in Eqs. (16.1)–(16.6), producing $\mathbf{h}_t$—the hidden state of the first layer. From Eq. (16.6) we see that each element of $\mathbf{h}_t$ is bounded in $(-1, 1)$. Hence, this first layer is mapping the standard basis of $\mathbb{R}^{137}$ to the hypercube $(-1, 1)^{512}$. Likewise, the second layer takes as input the first-layer hidden states in $(-1, 1)^{512}$ and maps it to $(-1, 1)^{512}$. The third layer does the same, but using the second-layer hidden states. We finally reach the softmax layer, which maps $(-1, 1)^{512}$ onto the positive face of the ℓ_1 unit-ball in $\mathbb{R}^{137}$. Finally, a sampling operation projects that point to an element of the standard basis of $\mathbb{R}^{137}$.

Each one of these vector spaces has significance with respect to the concepts learned by the model. The significance of the dimensions of the input and output spaces are clear since they are closely connected with the vocabulary we have designed: each orthogonal direction corresponds to one of the tokens. This fact helps us interpret those layers closest to the input and output, which are the first hidden layer and the softmax layer. Sturm [70] analyzes the parameters of the input layer of *folk-rnn (v2)* in terms of the vocabulary. It appears that the first hidden layer

has carved out subspaces of $(-1, 1)^{512}$ in which to represent the seven types of tokens. For instance, we see overlap in the representation of enharmonic pitches, such that their representation in the model is similar. Figure 16.9 shows the relationships between all pairs of pitch-token-related columns of $\mathbf{W}_{xc}$ in the first hidden layer. If two columns point in very similar directions, the color of the corresponding element of this matrix will be white. If they are orthogonal, the color will be gray. This comparison shows clear relationships between different tokens; e.g., pitch tokens 'A' and '=A' are encoded by this gate in nearly the same way, as are 'B' and '=B', as well as 'C' and '=C'. We also see a similarity between 'B', '_B' and '^A', which are the same in C mixolyidan, C dorian, and C minor. This shows *folk-rnn (v2)* has learned something about enharmonic relationships from the data alone, and that the cell gate of the first layer is treating these enharmonic pitch tokens in similar ways.

Sturm [69] analyzes the parameters of the softmax layer, which is transforming the hidden state of the third hidden layer into a probability distribution. This work shows how some principal directions are important for representing tokens of the measure type. This analysis also provides ways to adjust the behavior of the model, for instance, to make it less likely to output particular pitches. Much more analytical work has yet to be done to fully understand what is occurring in *folk-rnn (v2)*, but this kind of approach to analyzing an RNN is unique. The fact that the vocabulary of the system is not ambiguous helps to reveal the significance of particular subspaces.

16.4.2 Evaluation by Co-creation

One way of evaluating *folk-rnn* models is by looking at how composers can use them in the process of composition. Sturm, Ben-Tal and others have composed several pieces using *folk-rnn* [50]. One approach for a composer is to sift through generated outputs and locate ones that are interesting or inspire. A different approach involves searching the creative space [6] of *folk-rnn* transcriptions by iteratively generating transcriptions and changing parameters. In Boden's formulation [6], generative rules constrain the novel artifacts (poems, paintings, or music pieces, but also an idea or scientific discovery) that are possible to discover within a conceptual space. In that sense, *folk-rnn* is a manifestation of generative rules, which define the rather large conceptual space of all possible *folk-rnn* transcriptions. Iteratively generating outputs and tweaking the initialization parameters of the model is a search for valuable artifacts in that space. But, as we explain in more detail in Ben-Tal et al. [4], sifting for "gold" in this manner is not a straightforward process. The model is highly nonlinear, which could contribute to it producing interesting results, but also makes steering the generation process towards useful outputs somewhat unpredictable.

There are essentially three ways to interact with *folk-rnn* models. Changing the *random seed* of the sampling procedure just results in a different sampling from each posterior distribution. It has no musical significance. Changing the *temperature* parameter, which is the multiplicative factor T_s in (16.7), affects how "conservative" the sampling will be in each iteration. Figure 16.10 shows one example transcription

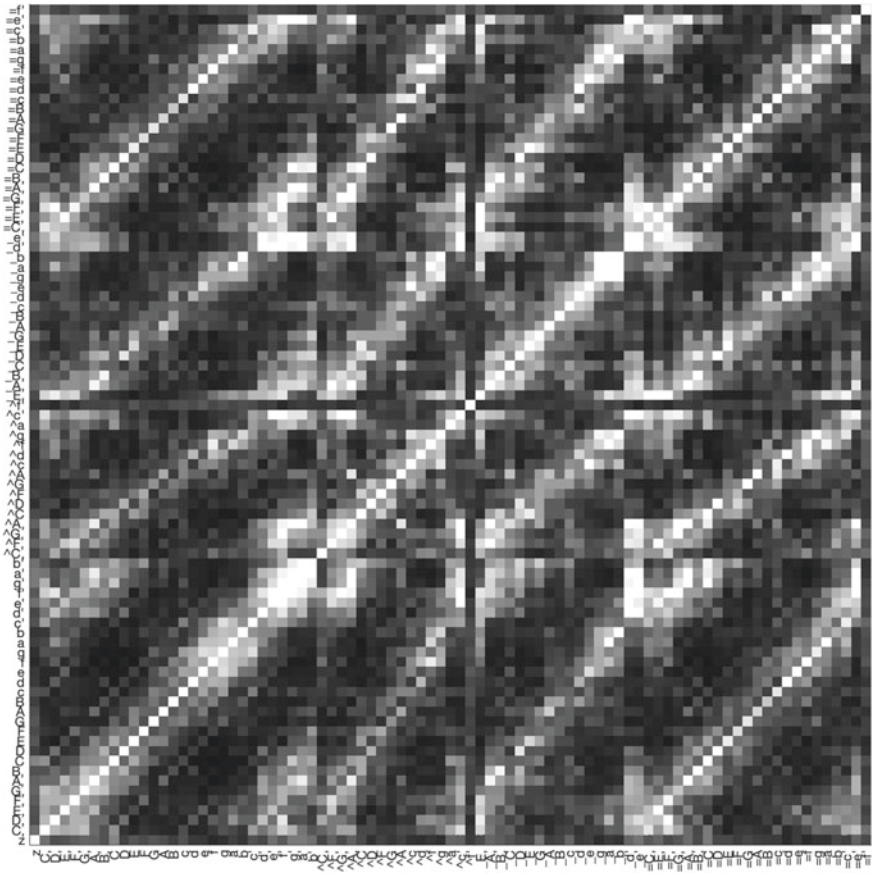

Fig. 16.9 Angles between columns cell matrix $\mathbf{W}_{xc}$ related to the pitch tokens in the first hidden-layer. White means the columns point in the same direction. The axes are labeled with the associated tokens. The diagonal structures show that columns of $\mathbf{W}_{xc}$ are related in ways that reflect enharmonic relationships; e.g., '`A`' and '`=A`' point in very similar directions; as do '`B`' and '`=B`'

generated by *folk-rnn (v2)* at a low temperature; and Fig. 16.11 shows an example generated at a high temperature. Setting the temperature to be very low will result in the network choosing the most likely event at each step. This can produce transcriptions having repeated notes and simple rhythms, but not always. High temperatures will result in transcriptions that adhere less to the conventions in the training data.

The third way a user can interact with *folk-rnn* is by giving it a sequence to continue. This provides perhaps the most immediate way to influence the content of the model output. Figure 16.12 shows how *folk-rnn (v2)* completes the given first measure '`M:4/4 K:Cmaj |: G C D E F G B A`' that is within the style of its training material. If a given sequence is a little outside the scope of what *folk-rnn* has seen it can produce unpredictable results. Figure 16.13 shows how *folk-rnn*

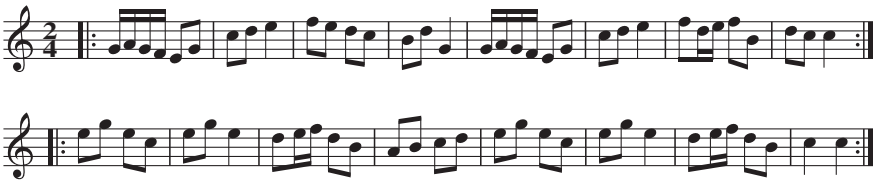

Fig. 16.10 Notation of a transcription generated by *folk-rnn (v2)* at a low sampling temperature ($T_s = 0.1$) The first part of the transcription is very close to a traditional Irish polka, "Babes in the Woods."

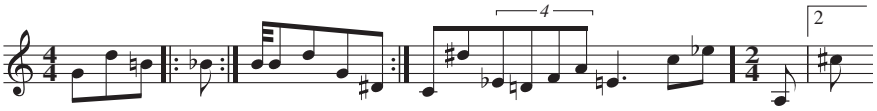

Fig. 16.11 Notation of a transcription generated by *folk-rnn (v2)* at a high sampling temperature ($T_s = 3$)

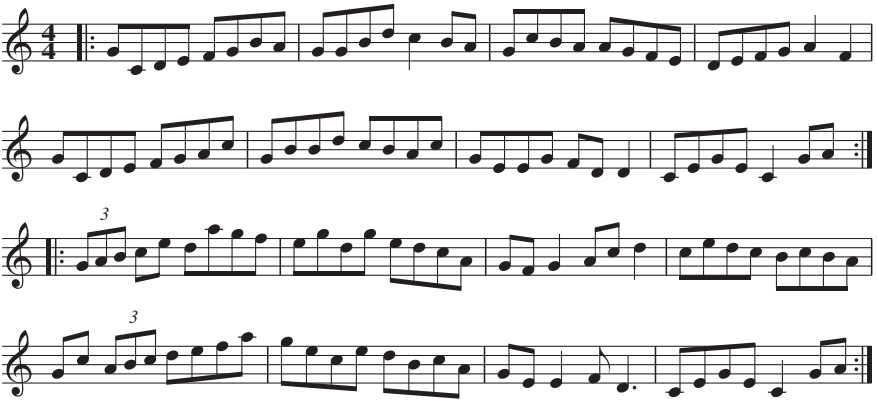

Fig. 16.12 Notation of a transcription generated by *folk-rnn (v2)* initialized with 'M:4/4 K:Cmaj |: G C D E F G B A'

(v2) continues nearly the same measure, changed in only one token to make it less conventional.

The design of the interaction with systems like *folk-rnn* needs considerable attention for them to serve as useful co-creative tools. To engage wider audiences in the potential for machine learning to stimulate music making, we created a pair of websites [4]: folkrnn.org and themachinefolksession.org. The first provides a web interface to use *folk-rnn* models for generating transcriptions. The second is a growing archive of transcriptions created by people using *folk-rnn*. At this time, folkrnn.org has been more successful than the archive. The interface for generating transcriptions continues to be used with several hundred individual users each month (with spikes following mention of the website in media or large events).

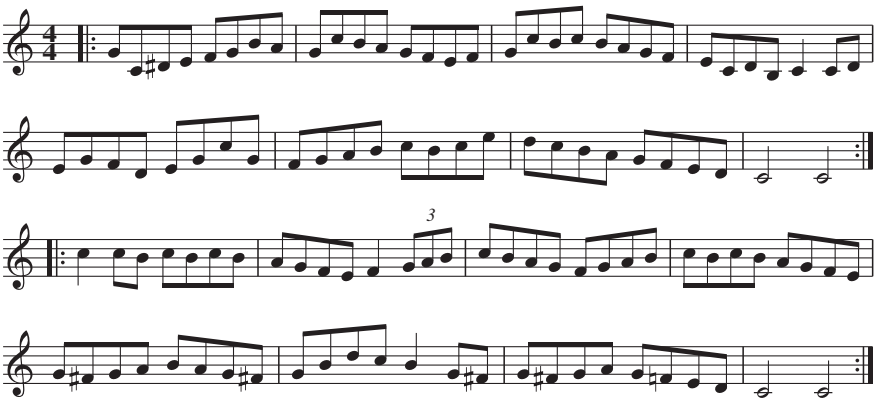

Fig. 16.13 Notation of a transcription generated by *folk-rnn (v2)* initialized with '`M:4/4 K:Cmaj |: G C ^D E F G B A`'. Compare to Fig. 16.12

https://themachinefolksession.org was intended to be a community portal for sharing interesting machine-generated but human-edited/performed tunes, but this has not gained much attention.

Finally, as noted above, the search for interesting or useful material from the generated outputs can be tedious. An "artificial critic" that sifts through generated material and identifies that having musical potential could greatly help—though a composer would like to be able to personalize "musical potential". More direct control over the features that a model learns, as well as the generation process, would also be useful. With increased knowledge about how the system learns and how it encodes its knowledge (see Sect. 16.4.1), it should be possible to provide additional methods of shaping the generated material.

16.4.3 Evaluation by Cherry Picking: "Let's Have Another Gan Ainm"

A different approach to gauging the creative potential of *folk-rnn* is to ask performers to make music out of the generated material. We collaborated with a range of musicians—both those familiar with the music traditions upon which *folk-rnn* was trained and musicians coming from other backgrounds. Many of these show a fair variety of results [71,72,74]. Significantly, most of the musicians did not have difficulties locating generated melodies they can perform, including performances on the soprano trombone, trumpet, and the double bass (instruments that are atypical in traditional music). At the same time, most of the musicians changed the generated melodies in performance. They frequently changed notes here and there, especially at the cadence points of phrases.

The relative ease of finding playable material in the generated outputs led us to record and release an album [72]. The aim was to investigate the potential of *folk-rnn* in creating material that sits comfortably within the domain of Irish traditional

music. We worked with Daren Banarsë, a London-based composer and performer who is well-versed in this kind of music. By his account, he looked at several hundred generated transcription and selected 20 to be recorded in the studio. Selecting the tunes involved judgments about what makes a tune 'good': does it work as a session tune? Is it playable by traditional instruments? Is it interesting and well-shaped? How well will it work with other tunes combined in a set? How well do the tracks add up to create a good album displaying both range and balance of the different dance types that constitute this musical world. Sturm and Ben-Tal [72] describes the process, and shows the changes Banarsë made to the generated material, and how he combined them with real traditional tunes to form the eleven tracks of the album.

We purposely kept secret the background of the album until a number of experts reviewed it. We did that to avoid bias in the reaction of the listener [55,63] rather than to discover if people would be fooled by the machine composed ones. The music was well-received by experts in this (fairly small) field—which is probably due in large part to the musical proficiency of the performers, and with Banarsë's careful curation of material. While cherry picking is an unacceptable way to evaluate many applications of machine learning, when applied to art it is not so different to how many artists work. Painters, writers, and composers constantly reject ideas in the creative process. In fact, a lack of ability to be self-critical can be a major hindrance. The creative process requires identifying weaknesses and fixing them, and persistence in doing that even when this can be slow and frustrating work. When it comes to music AI, the question to answer with cherry picking is not, "Which outputs best show the success of my model?" but, "How easy is it to find material generated by this model that I would like to work with?"

16.5 Ethical Considerations

A particularly original and illuminating outcome of this research has been the critical assessment of its motivations and impacts. We started a number of discussion threads on the forum of thesession.org requesting feedback on transcriptions generated by *folk-rnn (v2)*. The user `Ergo` agreed with another commenter about seeing no point to the research, but also mentioned some concern about its impact: "My concern is that some people, somewhere and sometime, may consider one or more of these tunes – maybe all of them? – to be actual traditional tunes…I think it's reckless to send 3,000 machine-created fiddle tunes into the world." Another user commented: "I would suggest confining your computerised efforts to the archives of whichever University you are at, I don't think this helps trad music in any way." In another thread, `Ergo` asks: "explain how this is going to contribute to [Irish traditional music]."

Someone later posted an article about our work to a Facebook group focused on Swedish folk music. Some comments among the 163 show real displeasure at the idea of involving computers in traditional music. One person writes, "Where is the heart and feeling in the computer?" Another writes

> Talk about soul-less tunes …MUSIC .. Especially folk music .. Must come from experiences, tradition's deep imprint. …Where the people are the focus, there the folk music characterizes the traditional cultural life. …When I see something like this, I get either pissed off or very critical.

Some express fears: "This takes away possibilities for real musicians to compose Music and earn a living!" Another writes

> You have stolen from existing music to offer a library based on goods made by [musicians] who have got their material from older musicians or composed their own now for tecknocrats within music to continue to steal patterns from the existing traditional music. …Within [pop music] there are rules for how much in seconds you are allowed to use from an existing piece of music for a mix or other use. One should pay the same respect to traditional music.

We experienced similar frictions when making the album, "Let's Have Another Gan Ainm" (Sect. 16.4.3) [72]. For instance, the professional musicians involved did not want to be too closely associated with the project. Though they were not a part of the research, and were only hired to perform on the album, they wanted to make sure that their professional careers were clearly separated.

Working together with Irish traditional harper Úna Monaghan also uncovered interesting aspects [74]. The music of this tradition is aural, and so modeling transcriptions is not really modeling the music. Irish traditional music is not a collection of transcriptions of music, but is bound together with functional notions, from dancing to making music together to expressing national pride [26,41,59]. Hence, anything produced by a music AI will be several steps away from the music that occurs in practice. Second, these AI-generated transcriptions, which necessarily come from a statistical mishmash of regional and historical styles, have only tenuous and confusing relation to the wider context that players use to perform this music. Because the data used for training *folk-rnn* is crowd-sourced, the choice of what to transcribe and how is not consistent in any manner across the corpus. What, therefore, should musicians do with these transcriptions? Should they try to correct or improve a generated transcription, to bring it "in line" with the tradition? Should they play them "straight", in tension to their own instinct and training?

These experiences show how our research can be seen in negative ways, and how our use of data could be an overstep. Our initial humorous application of machine learning could be regarded as trivializing a living tradition. While there is bound to be fear of human redundancy, or appeals to the narrative of machines taking control, many of the objections raised are more subtle and deserve careful attention. This motivated us to critically examine our assumptions, methodology, and research questions [43,75]. For example, since the data that we used to train music AI can be freely downloaded, does that give us the right to use it in the way we have? Considering that this data is essentially crowd-sourced over 18 years, contributed by people aiming to share, preserve,and advocate a particular form of traditional music, our use of the data for such a different outcome was likely unanticipated. That the dataset includes transcriptions of original works that are copyright protected can mean that our use of the data could be an infringement on those rights [75].

Our critical reflection has motivated us to learn much more about the living traditions from which we as researchers are benefiting, and to examine how our research could be detrimental and beneficial to Irish traditional music. While some of the musicians we worked with enjoyed learning the music, and some of the material has ended up in their regular repertoire [4], it is not completely clear how our research contributes to Irish traditional music. It did enable us to pay traditional musicians to perform and participate in various experiments. It also deepened Sturm's involvement with this music by attending traditional music summer schools in Ireland, and organizing regular learners' sessions in Stockholm. What is clear, however, is that this living tradition is not so weak that any music AI we or others might create can likely do any harm. One cannot confuse the tradition with symbols in a sequence, dots on a page, or tunes in an online database. Nonetheless, the variety of questions about the ethics of such research deserve to be discussed and assessed openly and regularly with practitioners.

16.6 Conclusion

Applying artificial intelligence to model and generate folk music offers many opportunities to study the capabilities and limitations of such methods, especially so in traditions that are living. It also motivates the critical examination of the use and misuse of artificial intelligence for music. In this chapter, we have surveyed the application of artificial intelligence to folk music. We have presented in depth our work in modeling transcriptions of traditional music from Ireland and Scandinavia using recurrent neural networks. We have also surveyed a variety of approaches we use to evaluate our models, from analyses of model parameters, to the use of the models in music creation. We finally discussed several contentious issues of our work, which motivates a careful study of its ethical dimensions.

Since our work with musicians and domain experts show that our machine learning models can generate transcriptions that are plausible within folk music traditions, it is clear that they have learned something relevant about identifiable and distinguishing characteristics from the training data. Special care needs to be taken, however. It is easy to fall into a trap of thinking human-like outputs from the machine reflect human-like learning or ability. Deeper examinations of our *folk-rnn* models reveal their knowledge about music to be very brittle. Nonetheless, we have found that these models can still be used as co-creative tools for music. In some cases, the brittleness of the knowledge of a model provides creative opportunities, which makes it a feature and not a bug. Indeed, our aims for building models of traditional music do not include generating a limitless number of cheap imitations. Instead, modeling folk music provides starting points to explore more interesting research questions.

One of these questions is the meaningful and effective evaluation of music AI and its involvement in music creation. While the field has progressed beyond simply listening to a few examples and confirming they sound reasonable, the evaluation of music-generating AI must include many dimensions, from the comparison of

statistical summaries of populations, to relating the models to music as a practice. Good evaluation methods identify ways for making improvements. We believe an essential component of the success of our project has been deep and meaningful partnerships between the engineering and musical sides, eventually engaging musicians in the research process and not just with the final outcome. The expertise of musicians, working in the specific domains from which we collect data for training AI—however superficially—is invaluable in that regard.

As to the future of *folk-rnn*, there are several directions we are exploring. We continue to analyze the parameters of our models to understand how they are encoding information, and how we might adjust them in musically meaningful ways; e.g., adjusting the parameters such that the model constructs jigs of nine-measures length instead of the conventional eight. Another direction is building an "artificial critic" that can streamline the search for interesting material a model has or could generate. This can be seen as a problem of information retrieval, for either an existing collection of material, or a collection that could be created by a given model. In line with this are methods for comparing collections of materials, including detecting plagiarism. All of these can potentially be incorporated into training models in more musically meaningful ways than just reproducing sequences of tokens.

Another avenue for future research can develop the system to accommodate polyphonic practices, or non-Western music. Polyphony means concurrent but also semi-independent voices, where the musical-logic has both a horizontal component (that governs the construction of each line) and a vertical one (that governs the interdependence of those lines). These different dimensions do not need to have the same or even similar rules. A challenge in applying machine learning to non-Western folk music entails finding a representation that is meaningful within the context of that tradition. Any representation necessarily abstracts away some aspects of the music, just as ABC notation does for Irish and Scandinavian folk music. The music AI researcher needs to produce a representation that can encode important and relevant aspects of music they want to model, and at the same time be aware of the aspects they discard.

References

1. ABCnotation. (2011). *Standard v2.1*. Retrieved May 30, 2020, from http://abcnotation.com/wiki/abc:standard:v2.1.
2. Ariza, C. (2009). The interrogator as critic: The Turing test and the evaluation of generative music systems. *Computer Music Journal*, *33*(2), 48–70.
3. Bacciu, D., Carta, A., & Sperduti, A. (2018). *Linear memory networks*. arXiv:1811.03356.
4. Ben-Tal, O., Harris, M. T., & Sturm, B. L. T. (2021). How music AI is useful: Engagements with composers, performers, and audiences. *Leonardo* (in press).
5. Bengio, Y., Boulanger-Lewandowski, N., & Pascanu, R. (2013). Advances in optimizing recurrent networks. In *IEEE International Conference on Acoustics, Speech, and Signal Processing* (pp. 8624–8628).

6. Boden, M. A. (2009). Computer models of creativity. *AI Magazine*, *30*(3).
7. Boulanger-Lewandowski, N., Bengio, Y., & Vincent. P. (2012). Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. In *Proceedings of International Conference on Machine Learning* (pp. 1159–1166).
8. Bronson, B. H. (1949). Mechanical help in the study of folksong. *Journal of American Folklore*, *62*, 81–86.
9. Brooks, F. P., Hopkins, A. L., Neumann, P. G., & Wright. W. V. (1957). An experiment in musical composition. *IRE Transactions on Electronic Computers*, *EC-6*(3), 175–182.
10. Chambers, J. (2014). *Chambers' Klezmer tunes*. Retrieved May 30, 2020, from http://trillian.mit.edu/~jc/music/abc/Klezmer/.
11. Chung, J., Gulcehre, C., Cho, K., & Bengio. Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. In *Proceedings of Neural Information Processing Systems*.
12. Cohen, J. E. (1962). Information theory and music. *Behavioral Science*, *7*(2), 137–163.
13. Collins, T., & Laney, R. (2017). Computer-generated stylistic compositions with long-term repetitive and phrasal structure. *Journal of Creative Music Systems*, *1*(2).
14. Colombo, F., Muscinelli, S. P., Seeholzer, A., Brea, J., & Gerstner, W. (2016). Algorithmic composition of melodies with deep recurrent neural networks. In *Proceedings of 1st Conference on Computer Simulation of Musical Creativity*.
15. Colombo, F., Seeholzer, A., & Gerstner, W. (2017). Deep artificial composer: A creative neural network model for automated melody generation. In: *Proceedings of EvoMUSART*.
16. Colombo, F., Brea, J., & Gerstner, W. (2019). Learning to generate music with Bachprop. In *Proceedings of Sound and Music Computing Conference*.
17. Colton, S., Charnley, J., & Pease, A. (2011). Computational creativity theory: The face and idea descriptive models. In *Proceedings of International Conference Computational Creativity*.
18. Conklin, D., & Witten, I. H. (1995). Multiple viewpoint systems for music prediction. *Journal of New Music Research*, *24*(1), 51–73.
19. Cope, D. (1991). *Computers and musical style*. Oxford University Press.
20. Dean, R., & McLean, A. (Eds.). (2018). *The Oxford handbook of algorithmic music*. Oxford, UK: Oxford University Press.
21. Dorronsoro, J. (1995). Bertso doinutegia. Euskal Herriko Bertsolari Elkartea. Retrieved May 30, 2020, from http://bdb.bertsozale.eus/es/.
22. Ebcioğlu, K. (1988). An expert system for harmonizing four-part chorales. *Computer Music Journal*, *12*(3), 43–51.
23. Eck, D., & Lapalme, J. (2008). *Learning musical structure directly from sequences of music*. Technical report, University of Montreal, Montreal, Canada.
24. Eck, D., & Schmidhuber, J. (2002). *A first look at music composition using LSTM recurrent neural networks*. Technical report, Instituto Dalle Molle di studi sull' intelligenza artificiale.
25. Eck, D., & Schmidhuber, J. (2002). Learning the long-term structure of the blues. In *Proceedings of International Conference on Artificial Neural Networks*, LNCS, Madrid, Spain (pp. 284–289).
26. Fairbairn, H. (1993). *Group playing in traditional Irish music: Interaction and heterophony in the session*. Ph.D. thesis, Cambridge University.
27. Ferentzy, E. N., & Havass, M. (1964). Human movement analysis by computer: Electronic choreography and music composition. *Computational Linguistics*, *3*, 129–188.
28. folk-rnn. (2017). *The folk-rnn session books*. Retrieved May 30, 2020, from https://highnoongmt.wordpress.com/2018/01/05/volumes-1-20-of-folk-rnn-v1-transcriptions/.
29. Foxley, E. (2001). *Eric Foxley's music database*. Retrieved May 30, 2020, from http://www.chezfred.org.uk/freds/music/database.htm.
30. Goel, K., Vohra, R., & Sahoo, J. K. (2014). Polyphonic music generation by modeling temporal dependencies using a RNN-DBN. In *Proceedings of International Conference on Artificial Neural Networks* (pp. 217–224).

31. Goienetxea, I., & Conklin, D. (2017). Melody transformation with semiotic patterns. In *Proceedings of International Symposium on Computer Music Multidisciplinary Research*.
32. Graves, A. (2013). *Generating sequences with recurrent neural networks*. arXiv:1308.0850.
33. Gray, R. (2017). The future of music: 'Bot Dylan' AI writes its own catchy folk songs after studying 23,000 tunes. *The Daily Mail*. Retrieved May 30, 2020, from https://www.dailymail.co.uk/sciencetech/article-4544400/Researchers-create-computer-writes-folk-music.html.
34. Hadjeres, G., Pachet, F., & Nielsen, F. (2017). DeepBach: A steerable model for Bach chorales generation. In *Proceedings of International Conference on Machine Learning* (pp. 1362–1371).
35. Hallström, E., Mossmyr, S., Sturm, B. L., Vegeborn, V. H., & Wedin, J. (2019). From jigs and reels to schottisar och polskor: Generating Scandinavian-like folk music with deep recurrent networks. In *Proceedings of Sound and Music Computing Conference*.
36. Havass, M. (1964). A simulation of musical composition. Synthetically composed folkmusic. *Computatiuonal Linguistics*, *3*, 107–128.
37. Herremans, D., Weisser, S., Sörensen, K., & Conklin, D. (2015). Generating structured music for bagana using quality metrics based on Markov models. *Expert Systems with Applications*, *42*, 7424–7435.
38. Hild, H., Feulner, J., & Menzel, W. (1992). HARMONET: A neural net for harmonizing chorales in the style of J. S. Bach. In *Proceedings of the Neural Information Processing Systems* (pp. 267–274).
39. Hiller, L. (1970). Music composed with computers—A historical survey. In *The computer and music*. Cornell University Press.
40. Hiller, L., & Isaacson, L. (1959). *Experimental music: Composition with an electronic computer*. New York, USA: McGraw-Hill Book Company.
41. Hillhouse, A. N. (2005). *Tradition and innovation in Irish instrumental folk music*. Master's thesis, The University of British Columbia.
42. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.
43. Holzapfel, A., Sturm, B. L., & Coeckelbergh, M. (2018). Ethical dimensions of music information retrieval technology. *Transactions of the International Society for Music Information Retrieval*, *1*(1), 44–55.
44. Johnson, D. D. (2017). Generating polyphonic music using tied parallel networks. In *Computational intelligence in music, sound, art and design* (pp. 128–143).
45. Jordanous, A. (2012). A standardised procedure for evaluating creative systems: Computational creativity evaluation based on what it is to be creative. *Cognitive Computation*, *4*(3), 246–279.
46. Karpathy, A. (2015). *The unreasonable effectiveness of recurrent neural networks*. Retrieved May 30, 2020, from http://karpathy.github.io/2015/05/21/rnn-effectiveness.
47. Karpathy, A., Johnson, J., & Li, F. (2015). *Visualizing and understanding recurrent networks*. arXiv:1506.02078.
48. Keith, J. (2020). *The session.org github repository*. Retrieved May 30, 2020, from https://github.com/adactio/TheSession-data.
49. Knees, P., & Schedl, M. (2016). *Music similarity and retrieval: An introduction to audio- and web-based strategies*. Springer.
50. Korshunova, I., Santos, J. F., & Sturm, B. L. T. (2015). *folk-rnn code repository*. Retrieved May 30, 2020, from https://github.com/IraKorshunova/folk-rnn.
51. Lapalme, J. (2005). Composition automatique de musique à l'aide de réseaux de neurones récurrents et de la structure métrique. Master's thesis, Université de Montréal.
52. Lerch, A. (2012). *An introduction to audio content analysis*. Wiley/IEEE Press.
53. Lieberman, F. (1970). Computer-aided analysis of Javanese music. In *The computer and music*. Cornell University Press.
54. LIVINGSTON. (2014). *Artificially intelligent folk songs of Canada*. Retrieved May 30, 2020, from http://www.folksingularity.com.

55. Moffat, D., & Kelly, M. (2006). An investigation into people's bias against computational creativity in music composition. In *Proceedings of Workshop on Computational Creativity*.
56. Mozer, M. C. (1994). Neural network composition by prediction: Exploring the benefits of psychophysical constraints and multiscale processing. *Cognitive Science*, *6*(2&3), 247–280.
57. Müller, M. (2015). *Fundamentals of music processing: Audio, analysis, algorithms, applications*. Springer.
58. Norbeck, H. (2020). *Henrik Norbeck's abc tunes*. Retrieved May 30, 2020, from http://www.norbeck.nu/abc/.
59. ÓhAllmhuráin, G. (1998). *A pocket history of Irish traditional music*. The O'Brien Press.
60. Olson, H. F., & Belar, H. (1961). Aid to music composition employing a random-probability system. *Journal of Acoustical Society America*, *33*(6), 862.
61. Paiement, J. F. (2008). *Probabilistic models for music*. Master's thesis, Ecole Polytechnique Fédérale de Lausanne (EPFL).
62. Pascanu, R., Mikolov, T., & Bengio, Y. (2013). On the difficulty of training recurrent neural networks. *Journal of Machine Learning Research*, *28*(3), 1310–1318.
63. Pasquier, P., Burnett, A., Thomas. N. G., Maxwell. J. B., Eigenfeldt, A., & Loughin, T. (2016). Investigating listener bias against musical metacreativity. In *Proceedings of International Conference on Computational Creativity*.
64. Pati, A., Lerch, A., & Hadjeres, G. (2019). Learning to traverse latent spaces for musical score in painting. In *Proceedings of International Symposium on Music Information Retrieval*.
65. Pease, A., & Colton, S. (2011). On impact and evaluation in computational creativity: A discussion of the Turing test and an alternative proposal. In *Proceedings of AISB Symposium on Musical Creativity*.
66. Pinkerton, R. C. (1956). Information theory and melody. *Scientific American*, *194*(2), 77–87.
67. Spiliopoulou, A., & Storkey, A. (2011). Comparing probabilistic models for melodic sequences. In *Proceedings Machine Learning Knowledge Discovery and Data* (pp. 289–304).
68. Sturm, B. L. T. (2015). *Eight short outputs ... (electroacoustic music composition)*. Retrieved May 30, 2020, from https://youtu.be/RaO4HpM07hE.
69. Sturm, B. L. T. (2018). How stuff works: LSTM model of folk music transcriptions. In *Proceedings of Joint Workshop on Machine Learning for Music, International Conference on Machine Learning*.
70. Sturm, B. L. T. (2018). What do these 5,599,881 parameters mean? An analysis of a specific LSTM music transcription model, starting with the 70,281 parameters of its softmax layer. In *Proceedings of Music Metacreation Workshop of International Conference on Computational Creativity*.
71. Sturm, B. L. T., & Ben-Tal, O. (2017). Taking the models back to music practice: Evaluating generative transcription models built using deep learning. *Journal of Creative Music Systems*, *2*(1).
72. Sturm, B. L. T., & Ben-Tal, O. (2018). *Let's have another Gan Ainm: An experimental album of Irish traditional music and computer-generated tunes*. Technical report, KTH Royal Institute of Technology.
73. Sturm, B. L. T., Santos, J. F., Ben-Tal, O., & Korshunova, I. (2016). Music transcription modelling and composition using deep learning. In *Proceedings of Conference on Computer Simulation of Musical Creativity*.
74. Sturm, B. L. T., Ben-Tal, O., Monaghan, U., Collins, N., Herremans, D., Chew, E., et al. (2018). Machine learning research that matters for music creation: A case study. *Journal of New Music Research*, *48*(1), 36–55.
75. Sturm, B. L. T., Iglesias, M., Ben-Tal, O., Miron, M., & Gómez, E. (2019). Artificial intelligence and music: Open questions of copyright law and engineering praxis. *MDPI Arts*, *8*(3).
76. Sundberg, J., & Lindblom, B. (1976). Generative theories in language and music descriptions. *Cognition*, *4*, 99–122.

77. Yang, L. C., & Lerch, A. (2018). On the evaluation of generative models in music. *Neural Computing and Applications*.
78. Yu, L., Zhang, W., Wang, J., & Yu, Y. (2016). *SeqGAN: Sequence generative adversarial nets with policy gradient*. arXiv:1609.05473.

Bob Sturm received a Ph.D. in Electrical and Computer Engineering from University of California, Santa Barbara, USA, in 2009. He has been employed as an academic at Aalborg University Copenhagen (Denmark), Queen Mary University of London (UK), and since 2018 at the Royal Institute of Technology (KTH), in Stockholm, Sweden. Since 2015, he has been building, learning from and collaborating with AI systems trained on transcriptions of traditional music. A portion of this work is supported by funding from the European Research Council under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 864189). E-mail: bobs@kth.se.

Oded Ben-Tal is a composer and researcher working at the intersection of music, computing, and cognition. His compositions include both acoustic pieces, interactive, live electronic pieces and multimedia work. As an undergraduate he combined composition studies at the Jerusalem Academy of Music with a B.Sc. in Physics at the Hebrew University, Israel. He did his doctoral studies at Stanford University's Center for Computer Research in Music and Acoustics (CCRMA). Since 2016 he is working on a multidisciplinary research project applying state of the art machine learning to music composition. He is a senior lecturer at the Performing Arts Department, Kingston University London, UK. E-mail: obental@gmail.com.