Comparing Models for Harmony Prediction in an Interactive Audio Looper

Benedikte Wallace and
Charles P. Martin

University of Oslo
RITMO Centre for Interdisciplinary
Studies in Rhythm, Time, and Motion
Department of Informatics {benediwa,charlepm}@ifi.uio.no

Abstract. Musicians often use tools such as loop-pedals and multitrack recorders to assist in improvisation and songwriting, but these tools generally don't proactively contribute aspects of the musical performance. In this work, we introduce an interactive audio looper that predicts a loop's harmony, and constructs an accompaniment automatically using concatenative synthesis. The system uses a machine learning (ML) model for harmony prediction, that is, it generates a sequence of chords symbols for a given melody. We analyse the performance of two potential ML models for this task: a hidden Markov model (HMM) and a recurrent neural network (RNN) with bidirectional long short-term memory (BLSTM) cells. Our findings show that the RNN approach provides more accurate predictions and is more robust with respect to changes in the training data. We consider the impact of each model's predictions in live performance and ask: "What is an accurate chord prediction anyway?"

Keywords: RNN · Deep Learning · Music Interaction · Machine Improvisation

1 Introduction

Though theoretically, there are no "wrong" chord choices, most people would agree that, for a given melody, some chord choices create a dissonance that does not sound particularly good. In order to decide precisely what chords should accompany a melody, human musicians apply knowledge from practical experience as well as formal rules of music theory. Stylistically appropriate chord choices rely on relationships between the melodic notes and chord, as well as temporal dependencies between chords in a sequence: what comes before and what comes next are both important. In this work[1] we apply machine learning (ML) to this problem in order to create a harmony prediction module for an interactive audio looping system. This predictive songwriting with concatenative accompaniment (PSCA) system uses a loop-pedal style interface to allow a musician to record a

[1] This research is an extension of the author's master's thesis [24].

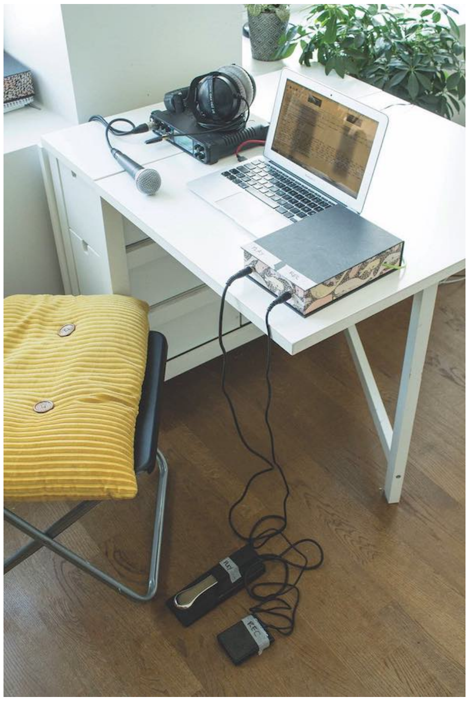


Fig. 1. Predictive Songwriting with Concatenative Accompaniment system setup: laptop running the PSCA software, sound card, microphone, headset and Arduino foot switch controller.

vocal melody, and then constructs an accompaniment layer of audio from their own recorded voice (Figure 1).

We consider two different models for harmony prediction in this research: a hidden Markov model (HMM) [18], the more traditional option, and a bidirectional recurrent neural network [21] with long short-term memory cells [10] (BLSTM), a newer deep neural network model. We analyse these two models from the perspective of predictive accuracy, and find that the BLSTM provides more accurate predictions overall, and is more robust to changes in the training data. Moreover, the BLSTM model provides more "creative" sounding chord sequences which can be further adjusted with a sampling temperature parameter. This makes the BLSTM network much more suitable for our interactive music tool. We discuss the implications of these results, particularly as they relate to the "accuracy" of a chord progression and suggest future directions for training musical machine learning systems to value creative choices.

1.1 The PSCA Looper

The underlying system developed for the PSCA is an audio looper written in Python and controlled by a pair of Arduino-interfaced foot switches. When the program is running the user sings into a microphone and controls recording and playback using the foot switches. The system allows the user to record, loop and play back layers of audio. In order to construct additional harmonies predicted by the models, the recorded audio is segmented, analyzed and added to a note bank according to its pitch. By concatenating and layering the necessary audio segments from the note bank the system is capable of adding new harmonies to the playback loop, creating an accompaniment which changes when the user records new layers. An overview of this process is shown in Figure 2.

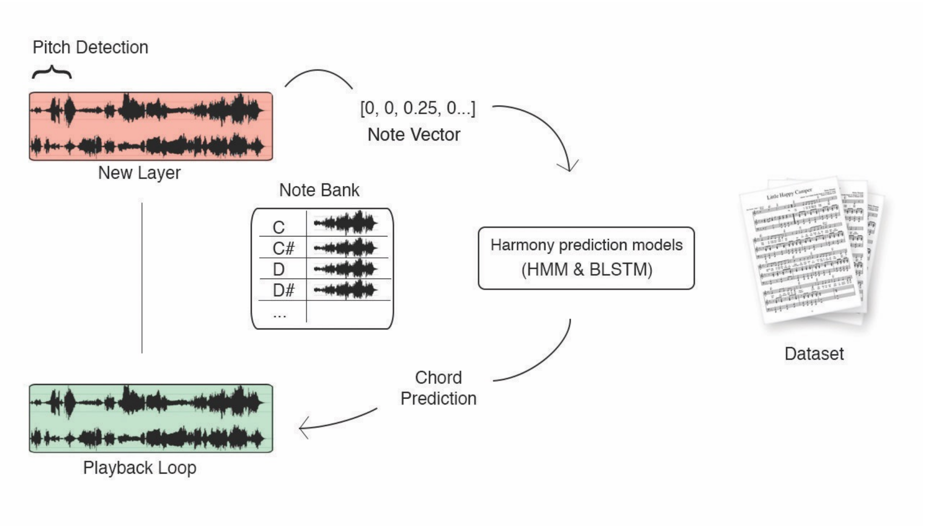


Fig. 2. PSCA system overview: Audio recorded by the user is added to the playback loop together with a chord selected by the harmony prediction models. The harmony is constructed using concatenated segments of the recorded audio.

1.2 Harmony Prediction

At the core of the PSCA system is a module for predicting suitable harmonies for a given sequence of melody notes. This module could be said to engage in *harmony prediction*, as shown in Figure 3. In this work, we consider the task of predicting suitable harmonies as a sequence-to-sequence prediction problem; given a input sequence of melodic notes, the model must choose a parallel sequence of harmonically appropriate chords. As shown in Figure 3 we restrict this problem to predicting just one chord for each bar of melody. A human would solve this problem by choosing a sequence of chords that works for the melody, as well as forming a sensible harmonic sequence. Our models will learn how to solve this problem through examples of chord and melody sequences from a data set of lead sheets. How well they perform in the harmony prediction task is evaluated using the accuracy of the predictions against the true values in the data set. Paired with qualitative analysis, the predictive accuracy gives us an idea of how good the models are.

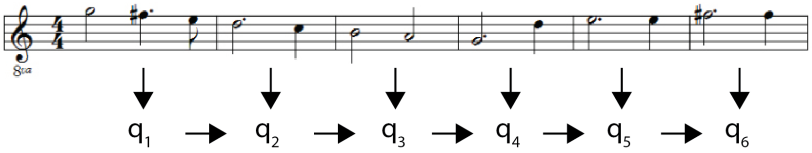


Fig. 3. The harmony prediction problem involves finding suitable chords $(q_1, \ldots, q_6)$ for a given melodic sequence. We consider the limited problem of finding one chord per bar of music.

1.3 Paper Overview

In the following section we will outline previous work on generating musical accompaniments using machine learning. Section 3 is devoted to our methods and materials, descriptions of the data set and our models as well as how these are used to facilitate harmony prediction in the PSCA looper. The results of model evaluation are shown in Section 4.1 and further discussed in Section 5.

2 Intelligent Loopers

Many musicians use looper pedals and effects to create solo performances with multiple layers of sound. These devices allow an initial phrase to be recorded which is then played back repeatedly and additional phrases can be added on top to build up a complex backing sound. While these devices are very popular, they can only play sounds that have been recorded. More intelligent looping devices have been proposed to overcome this limitation such as the "Reflexive Looper" [17] that modifies the looped phrases to fit a predetermined chord sequence. The Reflexive Looper determines the style of the musician's playing so that the generated accompaniment follows the musician's performance and can distinguish between several playing modes, filling the roles of drummer, bass player or keyboardist as needed.

SongSmith[22] is a system that automatically chooses chords to accompany vocal melodies. By singing into a microphone the user can experiment with different music styles and chord patterns through a user interface designed to be intuitive also for non-musicians. The system is based on a HMM trained on a music database consisting of 298 lead sheets, each consisting of a melody and an associated chord sequence.

JamBot [2] used two LSTM-RNNs. One predicts chord progressions based on a chord embedding similar to the natural language approach of word embeddings used in Google's word2vec [16], the other generates polyphonic music for the given chord progression. The results exhibit long term structures similar to what one would hear during a jam session. Martin and Torresen's RoboJam system [15] used a mixture density RNN (MDRNN) to generate additional layers for short touchscreen performances. This system learned to continue musical touchscreen control data, rather than high-level symbolic music.

Broadly, these systems for intelligently extending looped performances have used two ML architectures: HMMs, and RNNs with LSTM cells. HMMs were used in the successful SongSmith system and thus appeared to be a sufficient model for useful harmony prediction. LSTM-RNNs have been used for music generation [6], and were recently compared to HMMs for the specific task of harmony prediction, however they have rarely been used in interactive music applications [14]. Lim et al. [13] compared a HMM, a deep neural net HMM (DNN-HMM) and a bidirectional LSTM-RNN (BLSTM) for harmony prediction. Their findings suggested a large advantage in terms of accuracy for the BLSTM model over SongSmith's HMM. While using a BLSTM comes at a cost of computational power, such a model is still tractable on everyday computational devices.

Given the discrepancy in accounts for the success of HMM vs. BLSTM models, we decided to implement both architectures for our PSCA Looper. This has allowed us to gain insight into the accuracy of the models both through "accuracy" measures, as well as qualitatively through live performance and experimentation.

3 Building Models for Harmony Prediction

In this section we work towards creating two models of harmony prediction for use in the PSCA Looper using the HMM and BLSTM architectures respectively. While previous interactive systems had applied an HMM, the recent improvement of accessibility for RNNs demands that we consider both options.

3.1 Data Set

We used a data set of 1850 lead sheets in music XML format (.mxl) collected online. These contain both a melody and a corresponding sequence of chords. These were sourced from a data set previously shared at wikifonia.org until 2013. Each lead sheet consists of a monophonic melody and chord notation as well as key and time signature. The data set contains western popular music, with examples from genres like pop, rock, RnB, jazz as well as songs from musical theatre and children's songs. There are examples of songs in both major and minor keys. All songs originally in a major key were transposed to C major and songs in a minor key are transposed to A minor. Approximately 70% of the songs are classified as major keys and 30% as minor keys. Songs that contain key changes were removed from the data set.

The number of unique chords in the data set was 151; however, many of these chords featured very few examples. To reduce the number of chord choices, the data set was processed in two different ways. Our first set reduced each chord to one of five triad types—major, minor, suspended, diminished, and augmented—this ignores sevenths, as well as extended, altered, or added tones, leading to 60 possible chord types (5 triads times 12 root notes). The second approach uses only minor or a major triads, resulting in 24 chord types (2 chord types, 12 root notes). This 24-chord data set is more balanced and contains samples of each of the 24 chords. The 60-chord data set is much less balanced, with some chords having no samples at all; however, it allows for more varied harmonies to occur. The data set was further segmented into groups based on tonality with subsets corresponding to major and minor keys, as well as a mixed data sets.

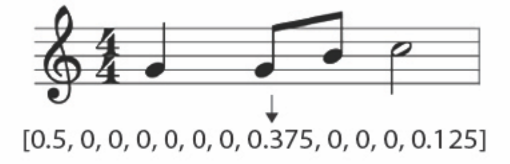


Fig. 4. Example of measure and the resulting note vector

The melody from each measure was transformed into a 12-dimension *note vector* representing the relative weight of each chromatic pitch class in that measure. The weights were calculated using duration of each pitch class normalized using the reciprocal of the song's time signature. An example of this transformation is shown in Figure 4.

The music21 toolkit [5] was used to process the data set and apply the note vector transformation. Each lead sheet was flattened into consecutive measures containing melody and chord information and information about key and time signature was extracted. In order to create example and target pairs for machine learning each measure in each of the songs should contain only one chord. For the songs where the occasional measure has more than one chord, all but the final chord in the measure are ignored. For measures with no chord notation, the chord from the preceding measure is repeated, and measures with no notes were ignored.

Training data for the ML models consisted of overlapping 8-measure sequences from the data set extracted with a step size of 1 measure. In the mixed data set there are 47564 chord and note vector pairs including "end token measures" that represent the end of the current song and the beginning of the next.

Their corresponding note vectors contain only zeros. An 8 measure long window slides over the data at 1 measure steps, creating a total of 47556 sequences containing 8 measures each.

3.2 Hidden Markov Models

HMMs have been applied to a range of sequence learning problems including speech processing [23,1]. HMMs model the relationship between a "hidden" underlying process that governs an observable sequence. The hidden process is assumed to conform to the Markov property, namely that future states are dependent only on the current state, and the observed emissions of each state occur independently of their neighbouring states. For the harmony prediction problem, the observable sequence is the notes found in each measure, represented by the note vectors described above. The process which governs these observable sequences, and which we want to model, is the chord progression. Training the HMM for the harmony prediction task consists of calculating the transition (A), start (π) and emission (B) probabilities for each state using the information found in the lead sheets. The start, transition and emission probabilities can be calculated directly by counting occurrences of chord transitions and the observed notes while traversing each measure of each lead sheet.

3.3 BLSTM

The bidirectional RNN [21], as the name implies, combines an RNN which loops through the input sequence forwards through time with an RNN which moves through the input from the end. Thereby, bidirectional networks can learn contexts over time in both directions. Separate nodes handle information forwards and backwards through the network. Thus, the output at time t can utilise a summary of what has been learned from the beginning of the sequence, forwards till time step t as well as what is learned from the end of the sequence, backwards till time step t. Bidirectional LSTM was presented by Graves and Schmidhuber in 2005 [9]. This architecture could be a better choice than a typical RNN for the PSCA, chord choice can be informed both by previous chords, as well as where the harmonic progressing is going next.

The hyperparameters and structure of the BLSTM are chosen to match those used by Lim et al. in their experiments [13]. As their data set and preprocessing choices are similar to ours it is assumed that these hyperparameters will yield similar results for our implementation. The input and output layers are time distributed layers, they apply the same fully-connected layer of weights to each time step. The input layer has 12 units which represent the 12 notes, and the output layer has one unit for each of the unique chords. The hidden layers are two bidirectional RNNs with 128 LSTM units each, with hyperbolic tangent activation and a dropout rate of 0.2 between them. Softmax activation is used on the output layer in order to generate the probabilities of each chord. Categorical cross entropy is used as the cost function and the Adam optimiser is used for training with parameters following Kingma et al. [11]. The BLSTM model is trained with a batch size of 512 and early stopping to end training validation accuracy does not improve for 10 epochs. The trained weights of the model that achieves the best validation accuracy is saved and can be loaded into the PSCA system to generate chord predictions on-the-fly. When the BLSTM model has been fitted to the data we can sample from the probability distribution of the softmax output layer to generate predictions.

3.4 Generating Harmony Prediction

In order to generate predictions from the trained HMM the Viterbi algorithm [8] is used to decide the most likely hidden sequence for a given sequence of observations. Given a model, $HMM = (A, B, \pi)$, consisting of the bigram for transitioning between chords (A), the emission probabilities (B), and the start probabilities (π), and a set of observations, O, the Viterbi algorithm begins by looking at the observations from left to right. For each observation, the max probability for all states is calculated. By keeping a pointer to the previously chosen state the algorithm can back trace through these pointers to construct the most likely path. Note that this process is deterministic, that is, a given observation sequence will always yield the same predicted hidden sequence.

Generating predictions from the BLSTM requires us to sample stochastically from the softmax output of the model at the final time step. If instead one were to choose the token with the highest probability each time, (greedy sampling) the generated chord progression can become uninteresting and repetitive. It is useful to be able to control the amount of randomness when sampling stochastically. This randomness factor is referred to as the sampling *temperature*. When choosing the next token from the softmax probability distribution the values are first reweighted using the temperature, creating a new probability distribution with more or less randomness. The higher the temperature, the higher the entropy of the resulting probability distribution.

Exactly what temperature to use when reweighting the softmax output depends heavily on the task at hand. During model evaluation a greedy sampling strategy was used, but when implementing a model for use in the PSCA system we experimented with different temperatures in order to generate less repetitive chord progressions (See Table 1).

Table 1. Example of predictions sampled at different temperatures

Original chord sequence	C	Em	Am	Am	C	Em	Am	Am
Sampled at high temp 1.1	Dm	B	G	Am	G	G	C	C
Sampled at low temp 0.1	C	G	Am	Am	G	Em	C	C

4 Evaluating the models

To evaluate the HMM and BLSTM models, we performed a quantitative analysis of the models using cross-validation, and a qualitative analysis on the confusion matrices of these generated models to explore the accuracy of their predictions. These results were compared with the accuracy of a naive predictor that simply outputs the major triad of the first detected note within each bar. Only 12 possible chords can be generated using this approach, regardless of the song key. Namely, the major triads: *C, C#, D, D#, E, F, F#, G, G#, A, A#, B.* The accuracy of this naive prediction approach is calculated by looking at the number of correct matches between the true chords and the chords chosen using only the first note in each measure, then normalised using the total number of measures in the data. The accuracy of the naive approach on the mixed data set is 26.97%.

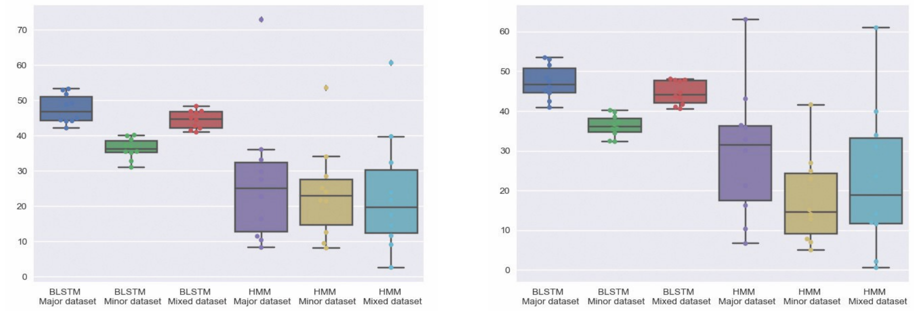
4.1 Cross Validation

Table 2. Average accuracy for k-fold cross validation with $k = 10$ for all six datasets, a naive predictor is included for comparison. The BLSTM performed best overall.

Data set	HMM (%)	BLSTM (%)	Naive (%)
Major 24-chord	27.10	47.56	
Minor 24-chord	23.43	36.30	
Mixed 24-chord	24.38	44.53	26.97
Major 60-chord	30.56	47.27	
Minor 60-chord	17.62	36.24	
Mixed 60-chord	23.70	44.60	

The HMM showed signs during training of being quite sensitive to the data it was trained on, resulting at times in extremely repetitive predictions. In order to examine this more closely we apply k-fold cross validation for both models and all data sets. When applying k-fold cross validation, the data is split into k sets, one set is used for testing the model's accuracy and the remaining $k - 1$ sets are used for training the model. This process is repeated for each of the k sets and the accuracy scores are averaged. The test was run on all versions of the data set, both the 24-chord and 60-chord version, as well as on the minor key, major key and mixed key versions (as described in section 3.1).

The results of the k-fold cross validation tests, with $k = 10$, are presented in Table 2. The BLSTM achieved higher average accuracy on all data sets. Using 24-chord classification in favour of 60-chord classification seems to result in some improvement for the HMM accuracy while the accuracy of the BLSTM predictions are strikingly similar regardless of the task requiring classification of 24 chord types, or using the imbalanced 60 chord type data set.



(a) K-fold cross validation accuracy 24-chord data set

(b) K-fold cross validation accuracy 60-chord data set

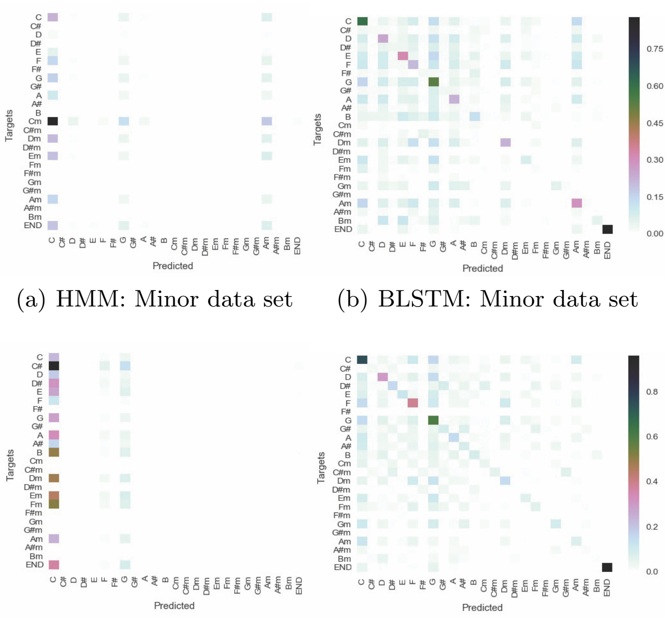
Fig. 5. K-fold cross validation accuracy using 24 and 60-chord data sets. The BLSTM model had the highest median accuracy over all tests, it also had a much narrower interquartile range than the HMM models, suggesting that it is less sensitive to changes in the data set.

The accuracy for each test has been plotted using combined swarm- and box plots in Figure 5. This shows that the accuracy of the HMM varies much more widely between folds than the accuracy of the BLSTM, regardless of using the 60-chord data sets or the 24-chord data set. This causes the interquartile range of the HMM results to be much wider, and the minimum and maximum accuracy vary between very low values—less than 10% accuracy on some folds—and higher values, over 60% accuracy, on others. This confirms the initial impressions that the HMM accuracy is strongly dependent on what segment of the data set was used to generate the transition, start and emission matrices. From the above results it is clear that the BLSTM is more robust across minor, major and mixed modes as well as having higher overall accuracy. The variations in HMM accuracy during *k*-fold cross validation shows how different segments of the data sets greatly affect its accuracy. In contrast, the BLSTM is more invariant to the different folds of the training data.

4.2 Creative Chord Predictions

Cross validation asks how accurate the chord choice was compared to the data set, but it is problematic that the naive predictor produces fairly good scores compared to the HMM model in particular. Figure 6 shows the confusion matrices for models trained on the 24-chord data set. We can see that the HMM models are heavily biased to return the basic chords I, IV, and V for major keys, and Im, V, VI for minor. In comparison, the BLSTM is not only more robust across minor and major tonalities, but also produces better predictions for other chords in the data set.

The model accuracy and confusion matrices confirm the findings of Lim et al. [13], in that the BLSTM outperforms the HMM as it is able to model the



(a) HMM: Minor data set (b) BLSTM: Minor data set

(c) HMM: Major data set (d) BLSTM: Major data set

Fig. 6. Confusion Matrix for HMM and BLSTM using data set with 24 unique chords. These results show that the HMM misclassifies most samples as belonging to classes C, G or Am while the BLSTM has better classifications for many other chords, shown by the clear diagonal line.

long term dependencies in western music more accuratley, allowing the current state to be affected by several preceding and following states. This point is also presented by Raczynski et al. [19] in their work on combining multiple probabilistic models in order to encompass several musical variables. Lim et al. also mention an imbalance in their data set: *"Moreover, the fact that the training data contains more frequent occurrences of C, F and G chords (over 60% in total samples) reduced the accuracy of the HMM model which uses the prior probability to obtain the posterior"* [13, p. 4]. The imbalance between classes C, G, F and the rest of the chords is mirrored in our data set as well. Our findings show that this imbalance also affects the BLSTM model, although not as severely. The predictions generated by the BLSTM model during k-fold validation shows how the BLSTM's predictions mirror the true chord distribution of the 24-chord data set. If almost 60% of our samples belong to classes C and G, the models can simply predict only these classes for all samples and achieve somewhat reasonable accuracy, a problem also known as the accuracy paradox.

There are several ways to combat the issue of imbalanced training data: Firstly of course, attempt to collect more data. Unfortunately, this is not a simple task due to the lack of a large, shared research database for popular music and jazz in lead sheet formats. An alternative could also be to generate more samples by writing music that contains the often unused chords, though naturally this would be a time-consuming task. A possible strategy would be to use Cohen's Kappa [4] in order to normalise classification accuracy using the imbalance of classes in the training samples. This approach may result in accuracy measurements that reflect the misclassifications of the minority classes better. An alternative option would be to use penalised models to impose additional cost on misclassification of a minority class. This can be used to 'force' the model to pay closer attention to the classes with the fewest training examples. Similarly, AdaCost [7], a variation of the AdaBoost method which uses the cost of misclassifications to update the training distribution on successive boosting rounds may also aid in reducing miscassification of minority classes.

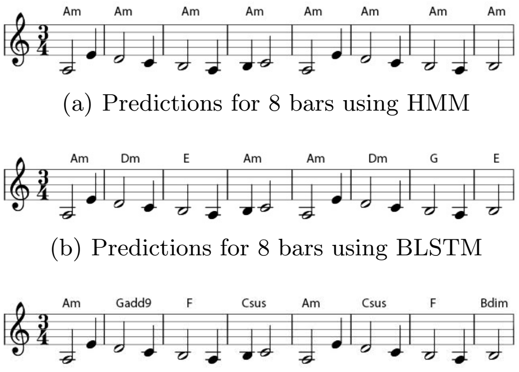
Resampling the data is also a common strategy for handling unbalanced data sets, and this is the strategy used for the PSCA as a starting point for improving our results. Resampling involves changing the data set either by adding samples to (oversampling) or removing (undersampling) from the training data. In this work, undersampling was used by splitting the data into minor and major sets. Also, undersampling was used to create a small, hand-picked data set for training the HMM used in the PSCA system.

5 What is an accurate chord prediction anyway?

It should be noted that the accuracy achieved by these models would not necessarily be considered good in other sequence prediction tasks. As mentioned in Section 1, there are few wrong answers when predicting chords for a given melody; several chords may sound equally good, and different listeners may appreciate different choices. Additionally, we consider predicted chords from both HMM and BLSTM models for a simple melody in Figure 7. While the BLSTM predicts chords that go well with the melody, its accuracy for this example is only 25%. This calls into question whether predictive accuracy is an appropriate measure of quality for harmony prediction.

ML models for the creative arts must take into account that the quality of a choice could depend on the observer, as pointed out by Schmidhuber [20]. Using classification accuracy as a metric for evaluating the models, or indeed training them, may therefore be misleading; rather, we should look at different approaches to measuring musicality. One option might be to include novelty, or curiosity [20], as a training heuristic, similar to those used in reinforcement learning [12][3]. For the BLSTM, this could be incorporated into the cost function used at each training step and help reward the model for making interesting and suitable, but not exactly correct, chord choices.

In fact, we took a heuristic version of this approach at the model selection level, by choosing models for the PSCA looper that produced the the most interesting chord choices. Rather than selecting the model with the highest accuracy we chose the model that had the most aesthetically pleasing results (to our ears). For the HMM an under-sampling technique was used, creating a small hand-



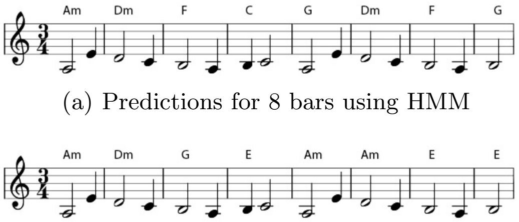
(a) Predictions for 8 bars using HMM

(b) Predictions for 8 bars using BLSTM

(c) Original

Fig. 7. Examples of different chord sequences as predicted by the models during k-fold cross validation using the 60-chord minor key data set, as well as the original chord sequence. While the BLSTM predicts suitable chords for each bar the HMM selects the root and does not generate a chord progression.

picked subset of the lead sheets. The BLSTM was trained using the 60-chord version of the minor-only data set and was encouraged to over fit to the training data slightly. We also used a higher sampling temperature during prediction to emphasize less likely chord choices. Examples of chord sequences produced by these two systems are shown in Figure 8.



(a) Predictions for 8 bars using HMM

(b) Predictions for 8 bars using BLSTM

Fig. 8. Examples of chord sequences as predicted by the hand-tuned models trained for use in the PSCA looper. Here, the HMM's result 8(a) has improved significantly.

These choices seem to be more artistically rewarding. In previous work [24], two musicians explored the PSCA looper comparing the heuristically selected BLSTM and HMM models[2]. Though the heuristic choices in training served to somewhat close the gap between the HMM and BLSTM performance (as exemplified in Figure 8), the BLSTM predictions were preferred by the participants

[2] Performance with the PSCA Looper (Direct download): https://goo.gl/59kVko

in most sessions. While rewarding curiosity has only been implemented heuristically so far, in future work we feel that BLSTM architecture with curiosity integrated into the cost function could lead to models which better learn to generate creative and interesting predictions for artistic tasks such as the harmony prediction.

6 Conclusion

In this paper we have compared two ML models of harmony prediction and discussed how they can be applied in an interactive audio looper for songwriting. The BLSTM produced better accuracy results during k-fold cross validation and exhibited more robustness against variations in the data. Although the predictions generated by the BLSTM were noticeably better than those generated by the HMM there was still an unwanted bias towards the majority classes C and G. We found that using the models that achieved the best accuracy in the PSCA would thereby lead to repetitive predictions with a strong major feel. This outcome was not desirable in the PSCA system. If the predicted chords are too repetitive this would be uninteresting for the performer. Similarly, predicting only major chords cause the harmonies to be quite limited. In order to mediate this issue a heuristic approach to model selection was taken, inspired by the concept of rewarding curiosity in chord prediction.

The PSCA system presents an example of an interactive system which uses machine learning techniques to enhance the musical experience of the user and has shown potential for commercial use and in live performance. In this work we have shown that finding a balance between predictions that are accurate (in reference to the training data) yet still interesting and creative is a core challenge of implementing ML in interactive music systems. Discovering efficient strategies for evaluating the models beyond their accuracy on the training data will therefor be an important direction for future research into interactive musical AI.

Acknowledgment

This work was supported by The Research Council of Norway as a part of the Engineering Predictability with Embodied Cognition (EPEC) project, under grant agreement 240862 and through its Centres of Excellence scheme, project number 262762.

References

1. Bahl, L.R., Jelinek, F., Mercer, R.L.: A maximum likelihood approach to continuous speech recognition. In: Readings in speech recognition, pp. 308–319. Elsevier (1990)
2. Brunner, G., Wang, Y., Wattenhofer, R., Wiesendanger, J.: Jambot: Music theory aware chord based generation of polyphonic music with LSTMs. In: 2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI). pp. 519–526. IEEE (2017). https://doi.org/10.1109/ICTAI.2017.00085

3. Burda, Y., Edwards, H., Pathak, D., Storkey, A., Darrell, T., Efros, A.A.: Large-scale study of curiosity-driven learning. In: Proceedings of the International Conference on Learning Representations (ICLR) (2019), https://arxiv.org/abs/1808.04355
4. Cohen, J.: A coefficient of agreement for nominal scales. Educational and psychological measurement **20**(1), 37–46 (1960). https://doi.org/10.1177/001316446002000104
5. Cuthbert, M.S., Ariza, C.: music21: A toolkit for computer-aided musicology and symbolic music data. In: Downie, J.S., Veltkamp, R.C. (eds.) Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR 2010). pp. 637–642. International Society for Music Information Retrieval, Utrecht, Netherlands (2010)
6. Eck, D., Schmidhuber, J.: Finding temporal structure in music: Blues improvisation with LSTM recurrent networks. In: Proceedings of the 12th IEEE Workshop on Neural Networks for Signal Processing. pp. 747–756. IEEE (2002). https://doi.org/10.1109/NNSP.2002.1030094
7. Fan, W., Stolfo, S.J., Zhang, J., Chan, P.K.: Adacost: misclassification cost-sensitive boosting. In: Proceedings of the Sixteenth International Conference on Machine Learning. ICML '99, vol. 99, pp. 97–105 (1999)
8. Forney, G.D.: The viterbi algorithm. Proceedings of the IEEE **61**(3), 268–278 (1973)
9. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Networks **5**(18), 602–610 (2005)
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
11. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
12. Lehman, J., Stanley, K.O.: Abandoning objectives: Evolution through the search for novelty alone. Evolutionary computation **19**(2), 189–223 (2011)
13. Lim, H., Rhyu, S., Lee, K.: Chord generation from symbolic melody using BLSTM networks. In: 18th International Society for Music Information Retrieval Conference (2017)
14. Martin, C.P., Ellefsen, K.O., Torresen, J.: Deep predictive models in interactive music. arXiv e-prints (Jan 2018), https://arxiv.org/abs/1801.10492
15. Martin, C.P., Torresen, J.: Robojam: A musical mixture density network for collaborative touchscreen interaction. In: Liapis, A., Romero Cardalda, J.J., Ekárt, A. (eds.) Computational Intelligence in Music, Sound, Art and Design. pp. 161–176. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-77583-8_11
16. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)
17. Pachet, F., Roy, P., Moreira, J., d'Inverno, M.: Reflexive loopers for solo musical improvisation. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 2205–2208. CHI '13, ACM, New York, NY, USA (2013). https://doi.org/10.1145/2470654.2481303
18. Rabiner, L., Juang, B.: An introduction to hidden Markov models. ASSP Magazine, IEEE **3**(1), 4–16 (1986). https://doi.org/10.1109/MASSP.1986.1165342
19. Raczyński, S.A., Fukayama, S., Vincent, E.: Melody harmonization with interpolated probabilistic models. Journal of New Music Research **42**(3), 223–235 (2013)

20. Schmidhuber, J.: Developmental robotics, optimal artificial curiosity, creativity, music, and the fine arts. Connection Science **18**(2), 173–187 (2006)
21. Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing **45**(11), 2673–2681 (1997)
22. Simon, I., Morris, D., Basu, S.: Mysong: Automatic accompaniment generation for vocal melodies. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 725–734. CHI '08, ACM, New York, NY, USA (2008). https://doi.org/10.1145/1357054.1357169
23. Tokuda, K., Zen, H., Black, A.W.: An hmm-based speech synthesis system applied to english. In: IEEE Speech Synthesis Workshop. pp. 227–230 (2002)
24. Wallace, B.: Predictive songwriting with concatenative accompaniment. Master's thesis, Department of Informatics, University of Oslo (2018)