ISOLATED INSTRUMENT TRANSCRIPTION USING A DEEP BELIEF NETWORK

Gregory Burlet, Abram Hindle
Department of Computing Science
University of Alberta
{gburlet,abram.hindle}@ualberta.ca

ABSTRACT

Automatic music transcription is a difficult task that has provoked extensive research on transcription systems that are predominantly general purpose, processing any number or type of instruments sounding simultaneously. This paper presents a polyphonic transcription system that is constrained to processing the output of a single instrument with an upper bound on polyphony. For example, a guitar has six strings and is limited to producing six notes simultaneously. The transcription system consists of a novel pitch estimation algorithm that uses a deep belief network and multi-label learning techniques to generate multiple pitch estimates for each audio analysis frame, such that the polyphony does not exceed that of the instrument. The implemented transcription system is evaluated on a compiled dataset of synthesized guitar recordings. Comparing these results to a prior single-instrument polyphonic transcription system that received exceptional results, this paper demonstrates the effectiveness of deep, multi-label learning for the task of polyphonic transcription.

1. INTRODUCTION

The process of automatic music transcription involves the transformation of an audio signal into a digitally encoded music score through an analysis of the frequency and rhythmic properties of the acoustical waveform. The input audio signal may consist of an aggregation of signals from several different instruments and may be monophonic or polyphonic. Though the transcription of monophonic musical passages is considered a solved problem [3], the transcription of polyphonic music "falls clearly behind skilled human musicians in accuracy and flexibility" [15].

In an effort to reduce the complexity, the transcription problem can be constrained by limiting the number of notes that sound simultaneously (polyphony), the genre of music being analyzed, or the number and type of instruments producing sound [2]. Imposing constraints on the domain of analyzed signals provides meaningful prerequisite knowledge to the transcription algorithm, allowing it to exploit certain properties of its input, consequently reducing the difficulty of transcription. With this in mind, the objective of this research is to improve the quality of transcriptions generated from isolated recordings of individual polyphonic instruments, such as the guitar, bass guitar, or piano.

A solution to the problem of isolated instrument transcription has substantial commercial interest with applications in musical games, instrument learning software, and music cataloguing. However, these applications seem far out of grasp given that the music information retrieval (MIR) research community has collectively reached a plateau in the accuracy of automatic music transcription systems [3]. In a paper addressing this issue, Benetos et al. [3] stress the importance of extracting expressive audio features and moving towards context-specific transcription systems. Also addressing this issue, Humphrey et al. [13, 14] propose that effort should be focused on audio features generated by deep belief networks instead of hand-engineered audio features, due to the success of these methods in other fields such as computer vision [18] and speech recognition [10]. The aforementioned literature provides motivation for investigating the viability of applying deep belief networks to the problem of isolated instrument transcription.

This paper presents a polyphonic transcription system containing a novel pitch estimation algorithm that addresses three arguable shortcomings in modern pattern recognition approaches to pitch estimation: first, the task of estimating multiple pitches sounding simultaneously is often approached using multiple one-versus-all binary classifiers [21, 22] in lieu of estimating the presence of multiple pitches using a single classifier; second, there exists no standard method to impose constraints on the polyphony of pitch estimates at any given time; and third, the discriminative power of latent audio feature representations, as produced by deep belief networks and autoencoders, are often overlooked in favour of more traditional features such as the short-time Fourier transform (STFT). In response to these points, the pitch estimation algorithm described in this work uses a *deep belief network* in conjunction with multi-label learning techniques to produce multiple pitch estimates for each audio analysis frame that conform to the polyphony constraints of the input instrument.

The structure of this paper is as follows: The subsequent section reviews algorithms for multiple fundamental frequency estimation, pitch detection, and note tracking. Section 3 describes the developed pitch estimation algorithm in the context of a larger polyphonic transcription system, which uses existing algorithms for note tracking. Section 4 presents a compiled ground-truth dataset of acoustic guitar recordings synthesized from crowdsourced tablature transcriptions that are then used to evaluate the

developed polyphonic transcription system. Section 5 ends with a discussion of the strengths and weaknesses of the transcription algorithm.

2. RELATED WORK

The first polyphonic transcription system for duets imposed constraints on the frequency range and timbre of the two input instruments as well as the intervals between simultaneously performed notes [20]. This work instigated a significant amount of research on this topic, which still aims to further the accuracy of transcriptions while gradually eliminating domain constraints.

In the infancy of the problem, polyphonic transcription algorithms relied heavily on digital signal processing techniques to uncover the fundamental frequencies present in an input audio waveform. To this end, several different algorithms have been proposed: perceptually motivated models that attempt to model human audition [16]; salience methods, which transform the audio signal to accentuate the underlying fundamental frequencies [17, 30]; iterative estimation methods, which iteratively select a predominant fundamental from the frequency spectrum and then subtract an estimate of its harmonics from the residual spectrum until no fundamental frequency candidates remain [17]; and joint estimation, which holistically selects fundamental frequency candidates that, together, best describe the observed frequency domain of the input audio signal [28].

The MIR research community is gradually adopting a machine-learning-centric paradigm for many MIR tasks, including polyphonic transcription. Several innovative applications of machine learning algorithms to the task of polyphonic transcription have been proposed, including hidden Markov models (HMMs) [23], non-negative matrix factorization [8,25], support vector machines [22], and artificial shallow neural networks [19]. Although each of these algorithms operate differently, the underlying principle involves the formation of a model that seeks to capture the harmonic, and perhaps temporal, structures of notes present in a set of training audio signals. The trained model then predicts the harmonic and/or temporal structures of notes present in a set of previously unseen audio signals.

Training a machine learning classifier for note pitch estimation involves extracting meaningful features from the audio signal that reflect the harmonic structures of notes and allow discrimination between different pitch classes. The obvious set of features exhibiting this property is the STFT, which computes the discrete Fourier transform (DFT) on a sliding analysis window over the audio signal. However, somewhat recent advances in the field of deep learning have revealed that neural networks with many layers of neurons can be efficiently trained [12] and form a hierarchical, latent representation of the input features [18].

Using a deep belief network (DBN) to learn alternate feature representations of DFT audio features, Nam et al. [21] exported these audio features and injected them into 88 binary support vector machine classifiers: one for each possible piano pitch. Each classifier outputs a binary class label denoting whether the pitch is present in a given audio analysis frame. Using the same experimental set up as Poliner and Ellis. [22], Nam et al. [21] noted that the learned features computed by the DBN yielded significant improvements in the precision and recall of pitch estimates relative to standard DFT audio features.

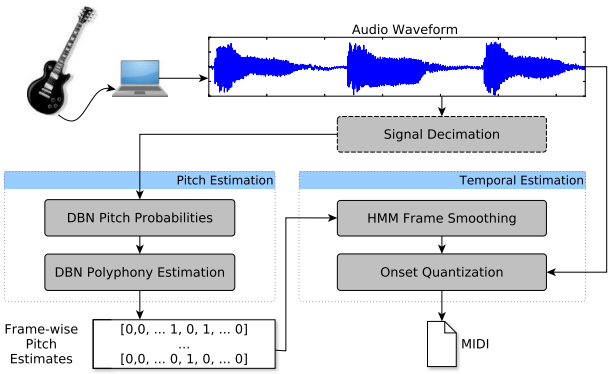

Figure 1. Workflow of the proposed polyphonic transcription algorithm, which converts the recording of a single instrument to a sequence of MIDI note events.

After note pitch estimation it is necessary to perform note tracking, which involves the detection of note onsets and offsets [2]. Several techniques have been proposed in the literature including a multitude of onset estimation algorithms [1, 9], HMM note-duration modelling algorithms [4, 24], and an HMM frame-smoothing algorithm [22]. The output of these note tracking algorithms are a sequence of note event estimates, each having a pitch, onset time, and duration. These note events may then be digitally encoded in a symbolic music notation for cataloguing or publishing.

3. ISOLATED INSTRUMENT TRANSCRIPTION

The workflow of the proposed polyphonic transcription algorithm is presented in Figure 1. The algorithm consists of an audio signal preprocessing step, followed by a novel DBN pitch estimation algorithm that conforms to the polyphony constraints of the input instrument. The note-tracking component of the polyphonic transcription algorithm uses a combination of the frame-smoothing algorithm developed by Poliner and Ellis [22] and the spectral flux onset estimation algorithm described by Dixon [9].

3.1 Audio Signal Preprocessing

The input audio signal is preprocessed before feature extraction. If the audio signal is stereo, the channels are averaged to produce a monaural audio signal. Then the audio signal is decimated to lower the sampling rate f_s by an integer multiple, $k \in \mathbb{N}^+$. Decimation involves low-pass filtering with a cut-off frequency of $f_s/2k$ Hz to mitigate against aliasing, followed by selecting every k^{th} sample from the original signal.

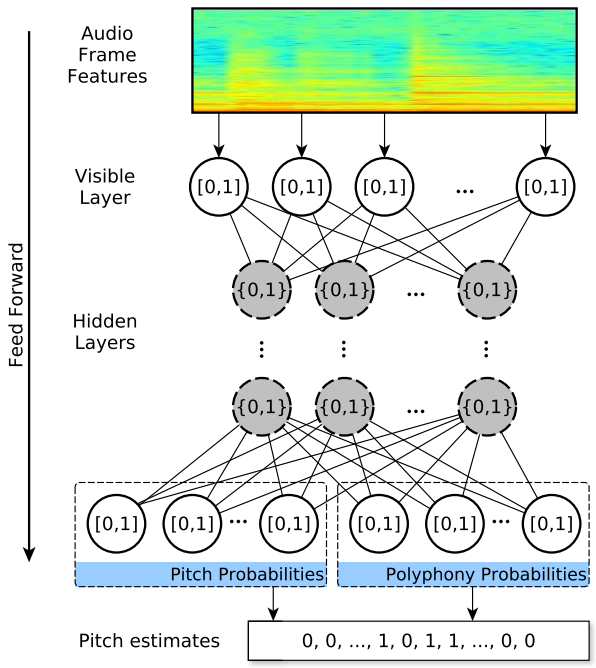

Figure 2. Structure of the deep belief network for note pitch estimation. Edge weights are omitted for clarity.

3.2 Note Pitch Estimation

The structure of the DBN pitch estimation algorithm is presented in Figure 2. The algorithm extracts audio features that are subsequently fed forward through the deep network, resulting in an array of posterior probabilities used for pitch and polyphony estimation.

First, features are extracted from the input audio signal. The power spectrum of each audio analysis frame is calculated using a Hamming window of size w samples and a hop size of h samples. Half of the spectrum is retained, resulting in $m = \lfloor w/2 \rfloor + 1$ features. The result is a matrix of normalized audio features $\Phi \in [0,1]^{n \times m}$, such that n is the number of analysis frames spanning the input signal.

The DBN consumes these normalized audio features; hence, the input layer consists of m nodes. There can be any number of stochastic binary hidden layers, each consisting of any number of nodes. The output layer of the network consists of $k + p$ nodes, where the first k nodes are allocated for pitch estimation and the final p nodes are allocated for polyphony estimation. The network uses a sigmoid activation as the non-linear transfer function.

The feature vectors Φ are fed forward through the network with parameters Θ, resulting in a matrix of probabilities $\mathrm{P}(\hat{Y}|\Phi,\Theta) \in [0,1]^{k+p}$ that is then split into a matrix of pitch probabilities $\mathrm{P}(\hat{Y}^{(pitch)}|\Phi,\Theta)$ and polyphony probabilities $\mathrm{P}(\hat{Y}^{(poly)}|\Phi,\Theta)$. The polyphony of the i^{th} analysis frame is estimated by selecting the polyphony class with the highest probability using the equation

$$\rho_i = \underset{j}{\operatorname{argmax}} \left(\mathrm{P}(\hat{Y}_{ij}^{(poly)}|\Phi_i,\Theta) \right). \qquad (1)$$

Pitch estimation is performed using a multi-label learning technique similar to the *MetaLabeler* system [26], which trains a multi-class classifier for label cardinality estimation using the output values of the original label classifier as features. Instead of using the matrix of pitch probabilities as features for a separate polyphony classifier, increased recall was noted by training the polyphony classifier alongside the pitch classifier using the original audio features. Formally, the pitches sounding in the i^{th} analysis frame are estimated by selecting the indices of the ρ_i highest pitch probabilities produced by the DBN. With these estimates, the corresponding vector of pitch probabilities is converted to a binary vector $\hat{Y}_i^{(pitch)} \in \{0,1\}^k$ by turning on bits that correspond to the ρ_i highest pitch probabilities.

For training and testing the algorithm, a set of pitch and polyphony labels are calculated for each audio analysis frame using an accompanying ground-truth MIDI file. A matrix of pitch annotations $Y^{(pitch)} \in \{0,1\}^{n \times k}$, where k is the number of considered pitches, is computed such that an enabled bit indicates the presence of a pitch. A matrix of polyphony annotations $Y^{(poly)} \in \{0,1\}^{n \times p}$, where p is the maximum frame-wise polyphony, is also computed such that a row is a one-hot binary vector in which the enabled bit indicates the polyphony of the frame. These matrices are horizontally concatenated to form the final matrix $Y \in \{0,1\}^{n \times (k+p)}$ of training and testing labels.

The deep belief network is trained using a modified version of the greedy layer-wise algorithm described by Hinton et al. [12]. Pretraining is performed by stacking a series of restricted Boltzmann machines and sequentially training each in an unsupervised manner using 1-step contrastive divergence [5]. Instead of using the "up-down" fine-tuning algorithm proposed by Hinton et al. [12], the layer of output nodes are treated as a set of logistic regressors and standard backpropagation is conducted on the network. Rather than creating features from scratch, this fine-tuning method is responsible for modifying the latent features in order to adjust the class boundaries [11].

The canonical error function to be minimized for a set of separate pitch and polyphony binary classifications is the cross-entropy error function, which forms the training signal used for backpropagation:

$$E(\Theta) = -\sum_{i=1}^{n}\sum_{j=1}^{k+p} Y_{ij} \ln \mathrm{P}(\hat{Y}_{ij}|\Phi_i,\Theta) + (1 - Y_{ij}) \ln(1 - \mathrm{P}(\hat{Y}_{ij}|\Phi_i,\Theta)). \qquad (2)$$

The aim of this objective function is to adjust the network weights Θ to pull output node probabilities closer to one for ground-truth label bits that are on and to pull probabilities closer to zero for bits that are off.

The described pitch estimation algorithm was implemented using the *Theano* numerical computation library for Python [6]. Computations for network training and testing are parallelized on the GPU. Feature extraction and audio signal preprocessing is performed using *Marsyas* [27].

3.3 Note Tracking

Although frame-level pitch estimates are essential for transcription, converting these estimates into note events with

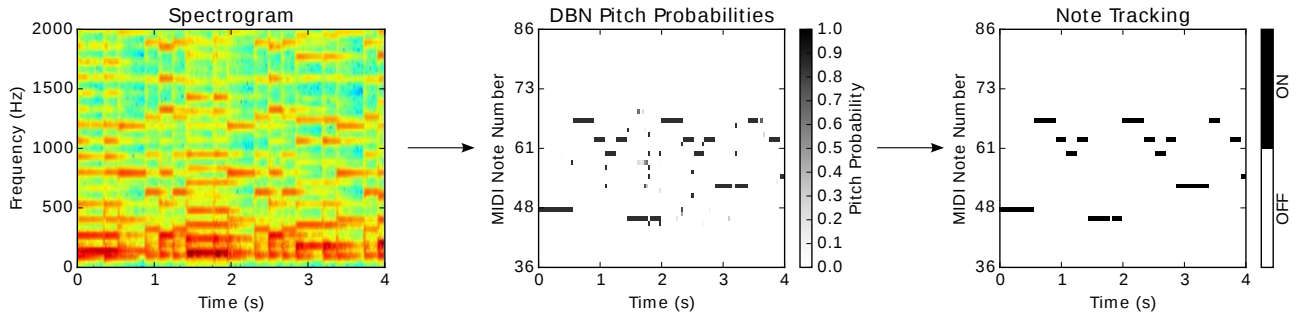

Figure 3. An overview of the transcription workflow on a four-second segment of a synthesized guitar recording.

an onset and duration is not a trivial task. The purpose of note tracking is to process these pitch estimates and determine when a note onsets and offsets.

3.3.1 Frame-level Smoothing

The frame-smoothing algorithm developed by Poliner and Ellis [22] is used to postprocess the DBN pitch estimates $\hat{Y}^{(pitch)}$ for an input audio signal. The algorithm allows a frame-level pitch estimate to be contextualized amongst its neighbours instead of solely trusting the independent estimates made by a classification algorithm.

Formally, the frame-smoothing algorithm [22] operates by training an HMM for each pitch. Each HMM consists of two hidden states: `ON` and `OFF`. The transition probabilities are computed by observing the frequency with which a pitch transitions between and within the `ON` and `OFF` states across analysis frames. The emission distribution is a Bernoulli distribution that models the certainty of each frame-wise estimate and is represented using the pitch probabilities $\mathrm{P}(\hat{Y}^{(pitch)}|\Phi, \Theta)$. The output of the Viterbi algorithm, which searches for the optimal underlying state sequence, is a revised binary vector of activation estimates for a single pitch. Concatenating the results of each HMM results in a revised matrix of pitch estimates $\hat{Y}^{(pitch)}$.

3.3.2 Onset Detection

If the HMM frame-smoothing algorithm claims a pitch arises within an analysis frame, it could onset at any time within the window. Arbitrarily setting the note onset time to occur at the beginning of the window often results in "choppy" sounding transcriptions. In response, the onset detection algorithm that uses spectral flux measurements between analysis frames [9] is run at a finer time resolution to pinpoint the exact note onset time. The onset detection algorithm is run on the original, undecimated audio signal with a window size of 2048 samples and a hop size of 512 samples. When writing the note event estimates as a MIDI file, the onset times calculated by this algorithm are used. The offset time is calculated by following the pitch estimate across consecutive analysis frames until it transitions from `ON` to `OFF`, at which point the time stamp of the end of this analysis frame is used. Note events spanning less than two audio analysis frames are removed from the transcription to mitigate against spurious notes.

Output of the transcription algorithm at each stage—from feature extraction to DBN pitch estimation to frame smoothing and quantization (note tracking)—is displayed in Figure 3 for a four-second segment of a synthesized guitar recording. The pitch probabilities output by the DBN show that the classifier is quite certain about its estimates; there are few grey areas indicating indecision.

4. TRANSCRIPTION EVALUATION

The polyphonic transcription algorithm is evaluated on a dataset of synthesized guitar tracks. Knowing that the input instrument is a guitar with six strings, the pitch estimation algorithm considers the $k = 51$ pitches from $C2$–$D6$, which spans the lowest note capable of being produced by a guitar in *Drop C* tuning to the highest note capable of being produced by a 22-fret guitar in *Standard* tuning.

Though a guitar with six strings is only capable of producing six notes simultaneously, a chord transition may occur within a frame and so the maximum polyphony increases above this bound. This is a side effect of a sliding-window analysis of the audio signal. The maximum frame-wise polyphony is calculated using the equation

$$p = \max_i \left(\left(Y^{(pitch)} \mathbb{1} \right)_i \right) + 1, \qquad (3)$$

where $\mathbb{1}$ is a vector of ones. The addition of one to the maximum polyphony is to accommodate silence where no pitches sound in an analysis frame.

4.1 Ground-truth Dataset

Using the methodology proposed by Burlet and Fujinaga [7], a ground-truth dataset of 45 synthesized acoustic guitar recordings paired with MIDI note-event annotations was compiled. The dataset was created by harvesting the abundance of crowdsourced guitar transcriptions uploaded to `www.ultimate-guitar.com` as tablature files that are manipulated by the *Guitar Pro* desktop application. [1] The transcriptions in the ground-truth dataset were selected by searching for the keyword "acoustic", filtering results to those that have been rated by the community as five out of five stars, and selecting those that received the most numbers of ratings and views. The dataset consists of songs by

[1] `www.guitar-pro.com`

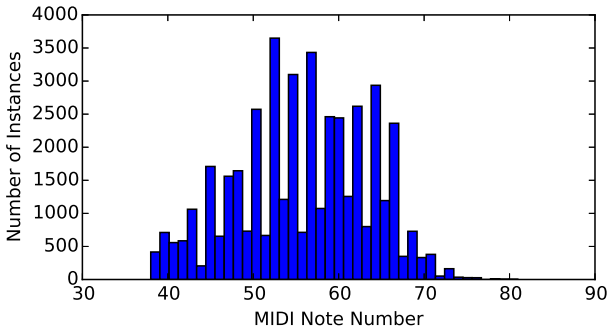

Figure 4. Distribution of note pitches in the ground-truth dataset.

artists ranging from The Beatles, Eric Clapton, and Neil Young to Led Zeppelin, Metallica, and Radiohead.

Each Guitar Pro file was preprocessed to remove extraneous instrument tracks other than guitar, removing repeated bars, and removing note ornamentations such as dead notes, palm muting, harmonics, pitch bends, and vibrato. The guitar model for note synthesis was set to a Martin & Co. acoustic guitar with steel strings and no capo. Finally, each Guitar Pro file is synthesized as a WAV file and also exported as a MIDI file, which captures the note events occurring in the guitar track. Recordings of real guitars would be ideal but the resources required to record and double-key annotate such a dataset is immense.

In total the dataset consists of approximately 104 minutes of audio, an average tempo of 101 beats per minute, 44436 notes, and an average polyphony of 2.34. The average polyphony is calculated by dividing the number of note events by the number of chords plus the number of individual notes. The distribution of note pitches in the dataset is displayed in Figure 4.

4.2 Frame-wise Pitch Estimation Evaluation

The songs in the compiled ground-truth dataset are partitioned into a training and testing set, such that roughly 80% of songs are allocated for training and 20% are allocated for testing. Several preliminary experiments with the proposed transcription system revealed that a sampling rate of 22050 Hz, a window size of 1024, a hop size of 768, a network structure of 350 nodes in the first three hidden layers followed by 1200 nodes in the penultimate layer yielded promising results. For network pretraining, 400 epochs were conducted with a learning rate of 0.05 using 1-step contrastive divergence with a batch size of 1000 training instances. For network fine-tuning, 30000 epochs were conducted with a learning rate of 0.05 and a batch size of 1000 training instances.

The frame-level pitch estimates computed by the DBN pitch estimation algorithm followed by the HMM frame-smoothing algorithm are evaluated using the following standard multi-label learning metrics [29]: precision (p), recall (r), f-measure (f), one error, and hamming loss. The one error provides insight into the number of audio analysis frames where the predominant pitch is estimated incorrectly. The hamming loss provides insight into the number of false positive and false negative pitch estimates across the audio analysis frames. In addition, the frame-level polyphony recall (r_{poly}) is calculated to evaluate the accuracy of polyphony estimates.

	r_{poly}	p	r	f	One Error	Hamming Loss
i.	0.52	0.66	0.60	**0.63**	0.22	0.04
ii.	0.52	0.72	0.69	**0.70**	0.18	0.03

Table 1. Frame-wise pitch estimation evaluation metrics: r_{poly} denotes the polyphony recall, p denotes precision, r denotes recall, and f denotes f-measure. The first row includes octave errors while the second row excludes them.

Using the ground-truth dataset, pretraining the DBN took 13 hours and fine-tuning took 9 hours using an Nvidia GPU with 1664 CUDA cores. After training, the network weights are saved so that they can be reused for future transcriptions. The results of the DBN pitch estimation algorithm are presented in Table 1. After HMM frame smoothing the results substantially improve with a precision of 0.79, a recall of 0.64, and an f-measure of 0.71 when considering octave errors.

The results reveal that the 52% polyphony estimation accuracy likely hinders the frame-wise f-measure of the pitch estimation algorithm. Investigating further, when using the ground-truth polyphony for each frame an f-measure of 0.68 is noted before HMM smoothing. The 5% increase in f-measure reveals that the polyphony estimates are close to their ground-truth value. With respect to the one error, the results reveal that the DBN's belief of the predominant pitch—the label with the highest probability—is incorrect in 22% of the analysis frames, which improves to 18% when not considering octave errors. With respect to the hamming loss, the results show that, on average, 4% of pitch estimates in an analysis frame are false positives or negatives. Additionally, the results reveal a 7% increase in f-measure when disregarding octave errors. Comparison of these results with a state-of-the-art transcription algorithm is performed in the following section.

4.3 Note Event Evaluation

After HMM smoothing the frame-level pitch estimates computed by the DBN, onset quantization is performed and a MIDI file, which encodes the pitch, onset time, and duration of note events, is written. An evaluation procedure similar to the music information retrieval evaluation exchange (MIREX) note tracking task is conducted using the metrics of precision, recall, and f-measure. Relative to a ground-truth note event, an estimate is considered correct if its onset time is within 250ms and its pitch is equivalent. The accuracy of offset times are not considered. A ground-truth note event can only be associated with a single note event estimate.

Table 2 presents the results of this evaluation on the guitar tracks in the testing dataset. Additionally, these guitar

	DBN Transcription			
	Precision	Recall	f-measure	Runtime (s)
i.	0.73	0.59	**0.65**	**49.15**
ii.	0.76	0.62	0.68	–
	Zhou and Reiss [30]			
	Precision	Recall	f-measure	Runtime (s)
i.	0.71	0.50	**0.56**	**203.32**
ii.	0.76	0.53	0.60	–

Table 2. Precision, recall, and f-measure evaluation of note events transcribed using the DBN transcription algorithm compared to the Zhou and Reiss [30] algorithm. The first row includes octave errors while the second row excludes them.

tracks are transcribed by the single-instrument polyphonic transcription algorithm proposed by Zhou and Reiss [30], which was evaluated in the 2008 MIREX and received an f-measure of 0.76 on a dataset of 30 synthesized and real piano recordings.

The transcription algorithm described in this paper resulted in a 9% increase, or a 16% relative increase, in f-measure relative to the transcription algorithm developed by Zhou and Reiss [30], and further, performed these transcriptions in a quarter of the time. This result emphasizes a lucrative property of neural networks: after training, feeding the features forward through the network is accomplished in a small amount of time. An analysis of the number of octave errors made by both algorithms reveals that the DBN transcription algorithm made a similar number of note octave errors as the digital signal processing transcription algorithm proposed by Zhou and Reiss.

A subjective, aural analysis of the guitar transcriptions reflects these results: the predominant pitches and temporal structures of notes occurring in the input guitar tracks are more or less maintained. Another remark on the transcriptions is that when several guitar strums occur quickly in succession, the DBN transcription algorithm often transcribes only the first chord and prescribes it a long duration. This is likely a result of the temporally "coarse" window size of 1024 samples or a product of the HMM frame-smoothing algorithm. A remedy for this issue is to lower the window size, which has an undesirable side-effect of lowering the frequency resolution of the DFT.

5. CONCLUSION

The developed polyphonic transcription algorithm is capable of forming discriminative, latent audio features that are suitable for quickly transcribing isolated instrument recordings. The algorithm workflow consists of audio signal preprocessing, feature extraction, a novel pitch estimation algorithm that uses multi-label learning techniques to enforce polyphony constraints, frame smoothing, and onset quantization. The generated note event transcriptions are digitally encoded as a MIDI file.

An evaluation of the frame-level pitch estimates generated by the deep belief network on a dataset of synthesized guitar recordings resulted in an f-measure of 0.71 after frame smoothing. An evaluation of the note events output by the entire transcription algorithm resulted in an f-measure of 0.65, which is 9% higher than the f-measure reported by a state-of-the-art, single-instrument transcription algorithm [30] on the same dataset. A threat to validity is the use of synthesized guitar signals for training and testing, which could be addressed by hiring guitarists and expert annotators to compile a dataset of real recordings.

There are several directions of future work to improve the accuracy of transcriptions. First, there are substantial variations in the distribution of pitches across songs, and so the compilation of more training data is expected to improve the accuracy of frame-level pitch estimates made by the DBN. Second, alternate methods could be explored to raise the accuracy of frame-level polyphony estimates, such as training a separate classifier for predicting polyphony on potentially different audio features. Third, an alternate frame-smoothing algorithm that jointly considers the probabilities of other pitch estimates across analysis frames could further increase pitch estimation f-measure relative to the HMM method [22], which smooths the estimates of one pitch across the audio analysis frames. Finally, it would be beneficial to investigate whether the latent audio features derived for transcribing one instrument are transferable to the transcription of other instruments.

The results of this work encourage the use of deep architectures such as belief networks or autoencoders to form alternative representations of industry-standard audio features for the purposes of instrument transcription. Moreover, this work demonstrates the effectiveness of multi-label learning for pitch estimation, specifically when an upper bound on polyphony exists.

6. ACKNOWLEDGEMENTS

Special thanks are owed to Ruohua Zhou and Joshua Reiss for the opensource implementation of their transcription algorithm evaluated in this work, as well as the individuals who uploaded manual tablature transcriptions to `www.ultimate-guitar.com`. This research was generously funded by an Alberta Innovates Technology Futures Graduate Student Scholarship.

7. REFERENCES

[1] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. B. Sandler. A tutorial on onset detection in music signals. *IEEE Transactions on Speech and Audio Processing*, 13(5):1035–1047, 2005.

[2] E. Benetos, S. Dixon, D. Giannoulis, H. Kirchhoff, and A. Klapuri. Automatic music transcription: Challenges and future directions. *Journal of Intelligent Information Systems*, 41(3):407–434, 2013.

[3] E. Benetos, S. Dixon, D. Giannoulis, H. Kirchoff, and A. Klapuri. Automatic music transcription: Breaking the glass ceiling. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 1002–1007, Porto, Portugal, 2012.

[4] E. Benetos and T. Weyde. Explicit duration hidden Markov models for multiple-instrument polyphonic music transcription. In *Proceedings of the International Conference on Music Information Retrieval*, pages 269–274, Curitiba, Brazil, 2013.

[5] Y. Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009.

[6] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference*, pages 3–10, Austin, TX, 2010.

[7] G. Burlet and I. Fujinaga. Robotaba guitar tablature transcription framework. In *Proceedings of the International Society for Music Information Retrieval*, pages 421–426, Curitiba, Brazil, 2013.

[8] A. Dessein, A. Cont, and G. Lemaitre. Real-time polyphonic music transcription with non-negative matrix factorization and beta-divergence. In *Proceedings of the International Society for Music Information Retrieval Conference*, Utrecht, Netherlands, 2010.

[9] S. Dixon. Onset detection revisited. In *Proceedings of the International Conference on Digital Audio Effects*, pages 133–137, Montréal, QC, 2006.

[10] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

[11] G. E. Hinton. Learning multiple layers of representation. *Trends in Cognitive Sciences*, 11(10):428–434, 2007.

[12] G. E. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

[13] E. Humphrey, J. Bello, and Y. LeCun. Moving beyond feature design: Deep architectures and automatic feature learning in music informatics. In *Proceedings of the International Society for Music Information Retrieval*, pages 403–408, Porto, Portugal, 2012.

[14] E. Humphrey, J. Bello, and Y. LeCun. Feature learning and deep architectures: New directions for music informatics. *Journal of Intelligent Systems*, 41(3):461–481, 2013.

[15] A. Klapuri. Automatic music transcription as we know it today. *Journal of New Music Research*, 33(3):269–282, 2004.

[16] A. Klapuri. A perceptually motivated multiple-F0 estimation method. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 291–294, New Paltz, NY, 2005.

[17] A. Klapuri. Multiple fundamental frequency estimation by summing harmonic amplitudes. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 216–221, Victoria, BC, 2006.

[18] H. Lee, R. Grosse, R. Ranganath, and A. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the International Conference on Machine Learning*, pages 609–616, Montréal, QC, 2009.

[19] M. Marolt. A connectionist approach to automatic transcription of polyphonic piano music. *IEEE Transactions on Multimedia*, 6(3):439–449, 2004.

[20] J. A. Moorer. *On the segmentation and analysis of continuous musical sound by digital computer*. PhD thesis, Department of Music, Stanford University, Stanford, CA, 1975.

[21] J. Nam, J. Ngiam, H. Lee, and M. Slaney. A classification-based polyphonic piano transcription approach using learned feature representations. In *Proceedings of the International Society for Music Information Retrieval*, pages 175–180, Miami, FL, 2011.

[22] Graham E. Poliner and Daniel P.W. Ellis. A discriminative model for polyphonic piano transcription. *EURASIP Journal on Advances in Signal Processing*, pages 1–9, 2006.

[23] C. Raphael. Automatic transcription of piano music. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 1–5, Paris, France, 2002.

[24] M. Ryynänen and A. Klapuri. Polyphonic music transcription using note event modeling. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 319–322, New Paltz, NY, 2005.

[25] P. Smaragdis and J. C. Brown. Non-negative matrix factorization for polyphonic music transcription. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 177–180, New Paltz, NY, 2003.

[26] L. Tang, S. Rajan, and V. K. Narayanan. Large scale multi-label classification via metalabeler. In *Proceedings of the International Conference on World Wide Web*, pages 211–220, Madrid, Spain, 2009.

[27] G. Tzanetakis and P. Cook. MARSYAS: A framework for audio analysis. *Organised Sound*, 4(3):169–175, 2000.

[28] C. Yeh, A. Roebel, and X. Rodet. Multiple fundamental frequency estimation and polyphony inference of polyphonic music signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1116–1126, 2010.

[29] M. Zhang and Z. Zhou. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1819–1837, 2014.

[30] R. Zhou, J. D. Reiss, M. Mattavelli, and G. Zoia. A computationally efficient method for polyphonic pitch estimation. *EURASIP Journal on Advances in Signal Processing*, 2009(729494):1–11, 2009.