

INVARIANCES AND DATA AUGMENTATION FOR SUPERVISED MUSIC TRANSCRIPTION

John Thickstun*

Zaid Harchaoui*

Dean Foster[†]

Sham M. Kakade*

* University of Washington, [†] Amazon

{thickstn, sham}@cs.washington.edu, zaid@uw.edu, dean@foster.net

ABSTRACT

This paper explores a variety of models for frame-based music transcription, with an emphasis on the methods needed to reach state-of-the-art on human recordings. The translation-invariant network discussed in this paper, which combines a traditional filterbank with a convolutional neural network, was the top-performing model in the 2017 MIREX Multiple Fundamental Frequency Estimation evaluation. This class of models shares parameters in the log-frequency domain, which exploits the frequency invariance of music to reduce the number of model parameters and avoid overfitting to the training data. All models in this paper were trained with supervision by labeled data from the MusicNet dataset, augmented by random label-preserving pitch-shift transformations.

Index Terms— music information retrieval, convolutional neural networks, invariances, learning

1. INTRODUCTION

The prominent success of deep learning in vision has popularized end-to-end learning approaches to supervised classification tasks. These methods depend upon large quantities of labeled training data. Many researchers have proposed supervised approaches to music transcription using synthesized training data, including recent MIREX participants [1, 2, 3]. While synthesized recordings provide an effectively infinite supply of labeled data, MIREX results suggest that models trained on synthetic data do not generalize well to human recordings.

There is a large and growing collection of human recordings annotated with labels suitable for supervision of music transcription [4, 5, 6, 7]. This prompts us to investigate models that can make effective use of this data. While the amount of available data is substantial, it is not infinite. Furthermore, frames of music sampled from an audio recording are more highly correlated than, for example, a pair of images from ImageNet. We therefore focus our attention on models and data augmentation techniques that incorporate prior knowledge of invariances in the problem domain to efficiently use the data.

Recent work shows that end-to-end architectures construct a first-layer feature representation that is qualitatively comparable to classical frequency filterbank transforms such

as the STFT or CQT [8, 7]. We will see that deep end-to-end models overfit to current datasets of human performances. This leads us to reconsider filterbanks as a low-level representation of musical audio. By replacing the first layer of an end-to-end network with a fixed filterbank transform, we dramatically reduce the number of model parameters and the attendant risk of overfitting.

The filterbank representation has a further advantage over end-to-end learning: its channels are ordered from low to high frequency. This order structure introduces a topology on the channel axis (the frequency domain) that motivates a convolutional architecture, analogous to how the Euclidean structure of \mathbb{R}^2 motivates the classic ConvNets used in computer vision. Constructing a ConvNet on the channels of a filterbank representation yields performance gains from parameter sharing that are not obviously replicable in an end-to-end architecture.

In this paper, we explore convolutional architectures that exploit the topological structure of a frequency filterbank representation. In Section 2 we discuss related work on music transcription. In Section 3 we will describe the network architectures of the models under consideration. In Section 4 we discuss optimization of these networks, including label-preserving transformations that augment the size of the training data. We present our qualitative results in Section 5 along with quantitative results on the MusicNet dataset and MIREX evaluation dataset.

2. RELATED WORKS

To the best of our knowledge, music transcription was first considered as a supervised learning problem in [9] using labels obtained from MIDI files to train an SVM on the spectrograms of synthesized recordings of these MIDIs. Subsequent work on supervised transcription improves upon these results in two directions: use of more sophisticated models, and construction of datasets of recorded human performances (as opposed to synthesized data).

Labels on music recordings are typically obtained in one of two ways. One approach is to perform music on instruments that are wired to record a MIDI transcription as they are played. These transcriptions are precise time-aligned labels for a performance. The commercially available Yamaha

Disklavier piano is wired in this way, and has led to the creation of datasets such as MAPS [5]. The second approach is to solve an alignment problem, warping a musical score onto a given recording. This alignment can be constructed using an optimal-alignment protocol, as in the SyncRWC [4], Lakh [6], and MusicNet [7] datasets. Or it can be constructed using information supplied by a human annotator, for example the Su dataset used for the MIREX evaluation [10].

The development of models for music transcription conceptually factors into two subproblems: acoustic modeling and time series prediction. In this paper, we focus on the acoustic modeling problem, as introduced in [9]. Recent developments in this area model the acoustics using deep neural networks [11, 12] or convolutional neural networks [13, 14, 7]. Some recent work explores hybrid models that combine a deep or convolutional acoustic model with a recurrent time-series model to jointly estimate transcriptions [15, 16].

Choosing an appropriate model for a supervised learning problem requires consideration of both the structure of the problem and the available data. A highly biased model can compensate for a smaller dataset at the risk of making overly powerful assumptions about the problem structure; a more general model requires more data to overcome variance. The frequency-invariance ideas that lead to the best models presented in this paper are anticipated in [16, 13, 14]. Our contribution is to demonstrate that this class of models represents a good bias-variance tradeoff for current datasets. Our dataset augmentation techniques are inspired by analogous transformations introduced by the vision community for image classification [17, 18] and extensions of these ideas to audio [19].

3. METHODS

Given an audio segment $\mathbf{x} \in \mathcal{X}$, we seek to predict the notes present at the midpoint of \mathbf{x} , which we encode as a binary label vector $\mathbf{y} \in \{0, 1\}^{128}$. We accomplish this task by learning a feature map $f_\theta : \mathcal{X} \rightarrow \mathcal{H}$, along with a multivariate linear regression to estimate $\hat{\mathbf{y}}$ given the learned representation $f(\mathbf{x})$. We consider variants of four network architectures f_θ for this purpose.

We take \mathbf{x} to be real-valued audio frame with values in the range $[-1, 1]$, sampled at 44.1kHz. We preprocess \mathbf{x} by the normalization $\mathbf{x} \mapsto \mathbf{x}/\|\mathbf{x}\|_2$; this can be interpreted physically as normalizing the audible volume of each frame. The first layer of every network considered in this paper is a strided convolution with a 4,096-sample receptive field and a 512-sample stride. We use a frame of 16,384 samples, resulting in $25 = (16384 - 4096)/512$ regions per frame.

The 16,384-sample frame size reflects a tradeoff between a shorter frame, which could miss important context for the classification task, and a longer frame, which has diminishing returns at computational cost. Very long frames grow the number of parameters in the model to the point of overfitting. The 512-sample stride is subject to a similar tradeoff.

3.1. Two layer networks

The simplest model we consider is a two layer network. For each region of the layer-one convolution we compute a filterbank representation of the input, creating a spectrogram representation \mathcal{H} at layer two. We perform linear classification on $\log \mathcal{H}$, the pointwise logarithm of the spectrogram. We consider several variants on the choice of filterbank below, as well as an end-to-end architecture where the filters are learned from data.

(Short-time Fourier transform) This is the classical filterbank consisting of Fourier coefficient magnitudes. We truncate the magnitude spectrum at 6kHz because we find that frequencies above this cutoff do not meaningfully improve classification accuracy.

(Log-spaced filterbank) This filterbank consists of 512 sine and cosine filters with logarithmically-spaced frequencies ranging from 50Hz to 6kHz. For each filter pair $\mathbf{w}_{k,\sin}, \mathbf{w}_{k,\cos}$, we compute inner products with the input region $\mathbf{x}_t \in [-1, 1]^{4096}$ and sum the square of these values, analogous to the STFT:

$$\text{filter}_k = (\mathbf{w}_{k,\sin}^T \mathbf{x}_t)^2 + (\mathbf{w}_{k,\cos}^T \mathbf{x}_t)^2.$$

(Windowed filterbank) Here we apply the cosine window $1 - \cos(t)$ to each filter in our filterbank. This combats the spectral leakage phenomenon caused by boundary effects introduced by the finite-window frequency analysis [20]. We will examine the effects of windowing on both the STFT and log-spaced filterbank.

(Learned filterbank) In this architecture, the filter coefficients \mathbf{w}_k are learned as parameters in the classifier optimization. This network is discussed at length in [7]. We will revisit these results in Section 5 and compare them to the hand-crafted filterbanks discussed above.

3.2. Three layer networks

A natural extension of the two layer networks discussed above is a three layer network with a fully connected layer interposed between the layer-one convolutions and the linear output layer. If we interpret the output of layer one as a spectrogram, then this intermediate layer captures non-linear relationships between features of this spectrogram. A filter in layer two might be sensitive to a particular chord, for example, or to a certain progression of notes. In Section 5 we report results for two three-layer networks, one trained with a fixed log-spaced, cosine-windowed filterbank at layer one, and the other trained end-to-end from the raw audio.

In this vein, it is possible to build much deeper models on top of either the raw audio or filterbank representation. These ideas are explored in [12]. However, as we will see in Section 5, simply building a deeper architecture does not appear to boost performance for the note classification task. Instead, we will turn to translation-invariant networks that introduce additional layers to capture specific invariances in the data.

3.3. Translation-invariant networks

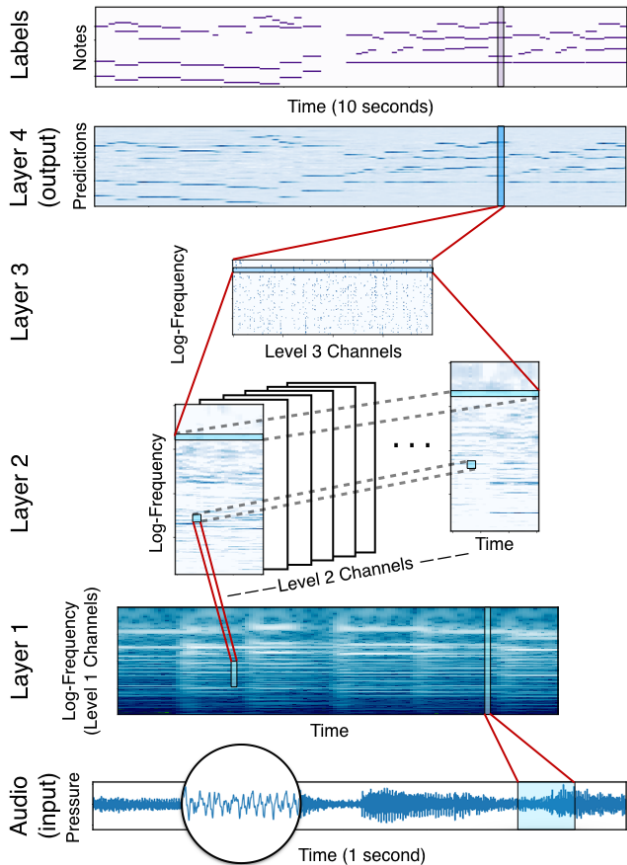


Fig. 1. A translation-invariant network for note classification. Audio input maps to Layer 1 according to the log-spaced, cosine-windowed filterbank described in Section 3.1. Layer 1 maps to Layer 2 by convolving a set of 128×1 learned filters along the log-frequency axis at each fixed time location. Layer 2 maps to Layer 3 by convolving again along the log-frequency axis, this time with a set of filters of height 1 that fully connect along the time and channel axes of Layer 2. Notes are predicted at Layer 4 by linear classification on the Layer 3 representation.

The translation-invariant network is built on top of a filterbank, with two learned representational layers. See Figure 1 for a visual schematic and description of this architecture. A handcrafted layer-one filterbank is crucial to support the translation-invariant filters at layer two. Because the filters are frequency-ordered, the layer-two filters can learn patterns that are invariant to translations in frequency. Consider, for example, a major triad chord. This pattern is preserved under linear translations in log-frequency space. In the translation-invariant architecture, a single filter consisting of only 128 parameters could be sensitive to major triads rooted at arbitrary frequencies. Compare this situation to a fully connected three

layer network (Section 3.2) which would require a separate filter to identify this chordal pattern at each location in the log-frequency spectrum.

The preceding arguments about music-theoretic concepts like intervals and triads are contingent on the use of a log-frequency filterbank for layer one. If we used a linear filterbank (for example, the STFT) then a linear shift in the frequency domain would correspond to a non-linear shift in the musical relationships between notes (low notes would translate further than high notes) due to the human ear’s logarithmic perception of frequency. On the other hand, the physics of audio (for example, overtones) exist on a linear scale. A model that uses a linear filterbank can exploit translation invariances in the physics. We find empirically that log-scale invariance yields greater performance gains for note classification than linear-scale invariance.

3.4. Channel convolutions

Finally, we consider an end-to-end network that uses the same architecture as the translation-invariant network, but treats the weights in the layer-one filterbank as optimization parameters (i.e. the filterbank is learned). Because the weights in layer one are learned from a random initialization, it is not clear that the learned filters will be ordered by frequency or that their output channels will exhibit any topological structure. However, because the layer-two convolutions in this architecture are designed to exploit local topological structure in the channels, we might hope that end-to-end parameter optimization would find a good topological structure for this space, a kind of self-organizing map [21].

4. TRAINING

We trained our models on the MusicNet dataset [7] with mini-batch stochastic gradient optimization (150 samples per mini-batch) using momentum ($\rho = .95$). The models are implemented in TensorFlow on an NVIDIA 1080Ti GPU. The final network weights are computed from a moving average of iterates with a decay factor of 2×10^{-4} .

Data augmentation. We augment our data by stretching or shrinking our input audio with linear interpolation. This corresponds to a pitch-shift in the frequency domain. For small shifts (± 5 semitones or less) the transformed audio sounds natural to the human ear. Randomly shifting each data point in a minibatch by an integral number of semitones in the range $[-5, 5]$ augments the dataset by an order of magnitude. And the translational nature of this augmentation reinforces the architectural structure of the translation-invariant networks described in Section 3. In addition to an integral semitone shift, we also apply a continuous shift to each data point in the range $[-.1, .1]$. This makes the models more robust to tuning variation between recordings.

5. RESULTS

Our best translation invariant network achieves 77.3% average precision on MusicNet, outperforming the previous state of the art reported in [12], and popular commercial software [22]. Furthermore, we find that the translation-invariant architecture substantially outperforms previously proposed end-to-end models on this dataset.

Model	Avg. Prec.	Acc.	Err.
filterbanks			
STFT (no compress)	40.4	15.9	.860
STFT	60.4	36.2	.681
log frequencies	62.7	39.8	.646
cosine windows	66.1	38.7	.637
log + windows	66.7	38.9	.633
three layer network	73.8	51.4	.541
end-to-end			
learned filterbank [7]	67.8	48.9	.634
three layer network	70.8	48.8	.558
deep complex [12]	72.9	-	-
channel convolution	73.3	50.4	.531
translation-invariant			
baseline	76.5	53.2	.496
pitch-shift	77.1	54.5	.482
wide layer 3	77.3	55.3	.474
commercial software			
Melodyne [22]	58.8	41.0	.760

Table 1. Average Precision, Accuracy, and Error for each of the models discussed in this paper, evaluated using the test set from [7]. Average Precision is computed by scikit-learn [23]; Accuracy and Error use mir_eval [24]. The Accuracy and Error scores are assume a global prediction threshold of 0.4.

Performance is highly sensitive to layer one. While a naive filterbank performs poorly, log-spaced, cosine-windowed filters approach the performance of a learned filterbank (see Table 1; compare “log + windows” to “learned filterbank”). Also, note the importance of the compressive non-linearity used for all models except the one marked “no compress.”

We consider three variants of the translation invariant architecture: a baseline, the same model optimized with pitch-shift transformations, and a model with a large number of hidden nodes at layer three (4,096 versus 256). Pitch-shifting is a clear improvement over the baseline. Increasing the number of nodes at level 3 is beneficial but requires dramatically more nodes for small performance gains; there may be an opportunity to model the time-domain structure captured in this layer more efficiently. We do not observe performance increases by adding nodes to layer two (we use just 128 layer-two filters) suggesting that the translation-invariant architecture efficiently captures frequency-domain structure.

The end-to-end architectures significantly underperform

the corresponding architectures with a handcrafted layer-one filterbank. Because filterbanks are essentially realizable in an end-to-end architecture (see [7]; Section 4.3) we infer that these optimizations have converged to either a local minimum or a saddle point. In particular, the learned layer-one weights of the channel convolution model exhibit some structure between neighboring filters, but not the global frequency ordering exhibited by the hand-crafted filterbanks.

Regarding the MIREX 2017 evaluation results, we remark that MHMTM1 [3] is a neural network trained on synthesized data. Several such models appeared in recent years at MIREX [1, 2, 3]. Because these networks can be fed an effectively infinite stream of training data, the efficiencies considered in this paper are not relevant and a wide variety of network architectures could fit well to the training data. The fact that these networks do not generalize well to benchmark data underscores the importance of training on datasets of human performances.

Model	Prec.	Rec.	Acc.	Etot
MIREX 2009 Dataset				
THK1	82.2	78.9	72.0	.316
KD1	72.4	81.1	66.9	.419
MHMTM1	72.7	78.2	65.5	.441
WCS1	64.0	80.6	59.3	.569
ZCY2	62.7	56.2	50.6	.601
Su Dataset				
THK1	70.1	54.6	51.0	.529
KD1	45.9	45.0	38.1	.745
WCS1	63.6	39.7	35.7	.700
MHMTM1	61.2	36.8	35.2	.676
ZCY2	40.9	28.2	26.2	.799

Table 2. MIREX 2017 results for the top 5 participants by accuracy in each category of the Multiple Fundamental Frequency Estimation challenge. THK1 is the wide layer 3 translation-invariant model described in this paper.

Conclusion. Regarding models: the best-performing translation invariant model naively boosts the number of features at level three to integrate temporal information. If we could construct an invariance (perhaps a scale- or elastic-invariance) in this layer, it might boost performance like the translation invariance at layer one. Regarding dataset augmentation, pitch-invariance is one of many possible label-preserving transformations. Effective noise injection remains an open problem. The authors tried adding Gaussian white noise and saw no performance gains. But we believe that a more realistic noise model could have a strong impact on transcription performance.

Acknowledgements. This work was supported by NSF Grant DGE-1256082, the Washington Research Foundation for innovation in Data-intensive Discovery, and the CIFAR program “Learning in Machines and Brains.”

6. REFERENCES

- [1] D. Troxel, "Music transcription with a convolutional neural network 2016," in *Music Information Retrieval Evaluation eXchange (MIREX)*, 2016.
- [2] M. Marolt, "Multiple fundamental frequency estimation & tracking submission for mirex 2016," in *Music Information Retrieval Evaluation eXchange (MIREX)*, 2016.
- [3] S. Mita, G. Hatanaka, A. Meneses, T. Nattapong, and M. Daiki, "Mirex 2017 : Multi-instrumental end-to-end convolutional neural network for multiple f0 estimation," in *Music Information Retrieval Evaluation eXchange (MIREX)*, 2017.
- [4] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "Rwc music database: Music genre database and musical instrument sound database," in *International Society for Music Information Retrieval (ISMIR)*, 2003.
- [5] V. Emiya, R. Badeau, and B. David, "Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle," in *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 2010.
- [6] C. Raffel, "Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching," PhD Thesis, 2016.
- [7] J. Thickstun, Z. Harchaoui, and S. Kakade, "Learning features of music from scratch," in *International Conference on Learning Representations (ICLR)*, 2017.
- [8] S. Dieleman and B. Schrauwen, "End-to-end learning for music audio," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014.
- [9] G. Poliner and D. P. W. Ellis, "A discriminative model for polyphonic piano transcription," in *EURASIP Journal on Applied Signal Processing*, 2007.
- [10] L. Su and Yi-Hsuan Y., "Escaping from the abyss of manual annotation: New methodology of building polyphonic datasets for automatic music transcription," in *International Symposium on Computer Music Multidisciplinary Research (CMMR)*, 2015.
- [11] J. Nam, J. Ngiam, H. Lee, and M. Slaney, "A classification-based polyphonic piano transcription approach using learned feature representations," in *International Society for Music Information Retrieval (ISMIR)*, 2011.
- [12] C. Trabelsi, O. Bilaniuk, D. Serdyuk, S. Subramanian, J. F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, and C. J. Pal, "Deep complex networks," in *Arxiv report: <https://arxiv.org/abs/1705.09792>*, 2017.
- [13] R. Bittner, B. McFee, J. Salamon, P. Li, and J. Bello, "Deep salience representations for f0 estimation in polyphonic music," in *International Society for Music Information Retrieval (ISMIR)*, 2017.
- [14] J. Pons and X. Serra, "Designing efficient architectures for modeling temporal features with convolutional neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.
- [15] S. Sigtia, E. Benetos, N. Boulanger-Lewandowski, T. Weyde, A. Garcez, and S. Dixon, "A hybrid recurrent neural network for music transcription," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.
- [16] S. Sigtia, E. Benetos, and S. Dixon, "An end-to-end neural network for polyphonic piano music transcription," in *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 2016.
- [17] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis," in *International Conference on Document Analysis and Recognition (ICDAR)*, 2003.
- [18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems (NIPS)*, 2012.
- [19] B. McFee, E. J. Humphrey, and J. P. Bello, "A software framework for musical data augmentation," in *International Society for Music Information Retrieval (ISMIR)*, 2015.
- [20] L. Rabiner and R. Schafer, "Introduction to digital speech processing," in *Foundations and trends in signal processing*, 2007.
- [21] T. Kohonen, "The self-organizing map," in *Proceedings of the IEEE*, 1990.
- [22] Celemony, "Melodyne," <http://www.celemony.com/en/melodyne/what-is-melodyne>.
- [23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in python," in *Journal of Machine Learning Research (JMLR)*, 2011.
- [24] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, and D. P. W. Ellis, "mir_eval: A transparent implementation of common mir metrics," in *International Conference on Music Information Retrieval (ISMIR)*, 2014.

A. EXTENDED STATISTICS

In this appendix, we reiterate our main results (Table 1) on an expanded test set. We observe that multiple data points taken from within one recording are highly correlated. Therefore, to get a representative sample of MusicNet, it is important to hold out a larger test set of recordings than the one introduced in [7]. For the results in Table 3, we compute statistics on the following extended test set (the first three recordings here are the ones from [7]):

- Bach’s Prelude in D major for Solo Piano. WTK Book 1, No 5. Performed by Kimiko Ishizaka. MusicNet recording id 2303.
- Mozart’s Serenade in E-flat major. K375, Movement 4 - Menuetto. Performed by the Soni Ventorum Wind Quintet. MusicNet recording id 1819.
- Beethoven’s String Quartet No. 13 in B-flat major. Opus 130, Movement 2 - Presto. Released by the European Archive. MusicNet recording id 2382.
- Bach’s Cello Suite No. 4 in E-flat major, Movement 6 - Gigue. Released by the European Archive. MusicNet recording id 2298.
- Bach’s Violin Partita No. 3 in E major, Movement 6 - Bourree. Performed by Oliver Colbenston. MusicNet recording id 2191.
- Beethoven’s Piano Sonata No. 30 in E major, Op. 109, Movement 2 - Prestissimo. Performed by Paul Pitman. MusicNet recording id 2556.
- Beethoven’s Wind Sextet in E-flat major, Op. 71, Movement 3 - Menuetto - Quasi Allegretto. Performed by the Skidmore Wind Ensemble. MusicNet recording id 2416.
- Beethoven’s Violin Sonata No. 10 in G major, Op. 96, Movement 3 - Scherzo: Allegro - Trio. Performed by the Irrera Brothers. MusicNet recording id 2628.
- Schubert’s Piano Sonata in C minor, D958, Movement 3 - Menuetto Allegro. Released by the Museopen organization. MusicNet recording id 1759.
- Haydn’s String Quartet in D major, Op. 645, Movement 3 - Menuetto Allegretto. Released by the Museopen organization. MusicNet recording id 2106.

B. CODE

Code for all the experiments presented in this paper is available online at <https://github.com/jthickstun/>

Model	Avg. Prec.	Acc.	Err.
filterbanks			
STFT (no compress)	40.5	17.4	.855
STFT	62.9	42.4	.634
log frequencies	64.3	44.5	.619
cosine windows	66.1	44.1	.618
log + windows	66.4	44.4	.618
three layer network	76.3	55.4	.492
end-to-end			
three layer network	72.9	53.0	.503
channel convolution	74.6	54.0	.483
translation-invariant			
baseline	77.8	57.7	.449
pitch-shift	79.5	59.2	.432
wide layer 3	79.9	59.9	.423
commercial software			
Melodyne [22]	57.9	39.5	.744

Table 3. Average Precision, Accuracy, and Error for each of the models discussed in this paper, evaluated using the extended test set described in this appendix. Average Precision is computed by scikit-learn version 0.19.1 [23] (please note that older versions of scikit-learn contained a bug in the average precision metric implementation; see the release notes for version 0.19.1; all average precision numbers in this paper are computed using the implementation in version 0.19.1). Accuracy and Error use `mir_eval` [24]. The Accuracy and Error scores are assume a global prediction threshold of 0.4.

`thickstun2018invariances/`. The preprocessed version of MusicNet used in these experiments is available at http://homes.cs.washington.edu/~thickstn/icassp_data.tar.gz (11Gb download) and pre-trained weights for each model are available at http://homes.cs.washington.edu/~thickstn/icassp_weights.tar.gz (1.1Gb download). Further instructions on using this code and data can be found in the git repository’s README.md document.