INVARIANCES AND DATA AUGMENTATION FOR SUPERVISED MUSIC TRANSCRIPTION

John Thickstun⋆ Zaid Harchaoui⋆ Dean P. Foster† Sham M. Kakade⋆

⋆ University of Washington, † Amazon
{jthickstun,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 for supervised classification tasks. This approach depends upon large quantities of labeled training data. Many researchers have proposed supervised methods for music transcription that use 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 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 topological structure of a frequency filterbank representation. In Section 2 we discuss related work on music transcription. In Section 3 we 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 qualitative results in Section 5 along with quantitative results on the MusicNet dataset and MIREX evaluation dataset. Code for all the experiments presented in this paper is available online at https://github.com/jthickstun/thickstun2018invariances/.

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 MIDI. 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 lead 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 \( f_0 : \mathcal{X} \rightarrow \mathcal{H} \), along with a multivariate linear regression to estimate \( \mathbf{y} \) given the learned representation \( f(\mathbf{x}) \). We consider variants of four network architectures \( f_0 \) 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 striated 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 interpolated 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 $128 \times 1$ 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 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 minibatch stochastic gradient optimization (150 samples per minibatch) 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 $2 \times 10^{-4}$.

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 ($\pm 5$ semitones or less) the transformed audio sounds natural to the human ear. Randomly shifting each data points 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 $[-0.1, 0.1]$. 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.

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

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-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 only one of many label-preserving transformations. Effective noise injection remains an open problem; the authors tried Gaussian noise and saw no improvement. But we believe that a more realistic noise model could greatly improve transcription performance.

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.

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

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 only one of many label-preserving transformations. Effective noise injection remains an open problem; the authors tried Gaussian noise and saw no improvement. But we believe that a more realistic noise model could greatly improve 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.” We also thank NVIDIA for their donation of a GPU.
6. REFERENCES


