MELODY EXTRACTION USING A HARMONIC HARMONIC CONVOLUTIONAL NEURAL NETWORK

Jiří Balhar
Charles University
Institute of Formal and Applied Linguistics
balhar.j@gmail.com

Jan Hajič jr.
Charles University
Institute of Formal and Applied Linguistics
hajicj@ufal.mff.cuni.cz

ABSTRACT

Melody extraction is arguably one of the most challenging and potentially rewarding problems in Music Information Retrieval. It is melody that we are likely to recall after listening to a song and so it is one of the most relevant aspects of music. However the presence of accompaniment in songs makes the task hard to address using rule-based methods. During the last years data-driven methods based on deep learning started to outperform methods traditionally used in the field. In this paper we continue in these efforts and propose a new method for melody extraction. An architecture called *Harmonic Convolutional Neural Network*, based on a modification of convolutional neural networks to better capture harmonically related information in an input spectrogram with logarithmic frequency axis, was able to achieve state-of-the-art performance on most publicly available melody datasets.

1. INTRODUCTION

In the recent years the approaches to melody extraction have shifted to the use of deep learning (DL). These new methods transform the input signal to a time-frequency representation and then use DL techniques to sequentially process the transformed input. The result is a saliency map from which they select the melody f_0 trajectory and determine voicing. Among new DL-based works we can find variety of approaches that try new signal transformations or network topologies; on the other hand the building blocks out of which these systems are built are standard (fully connected layers, CNNs and RNNs) [1, 3, 7, 10]. In this paper we propose a new kind of CNN layer specialized for processing harmonic signals in audio. [1]

1.1 The use of CNNs for Melody Extraction

A harmonic signal is usually comprised of a fundamental frequency and overtones that are spaced apart by constant ratios. These features are therefore non-local on a input

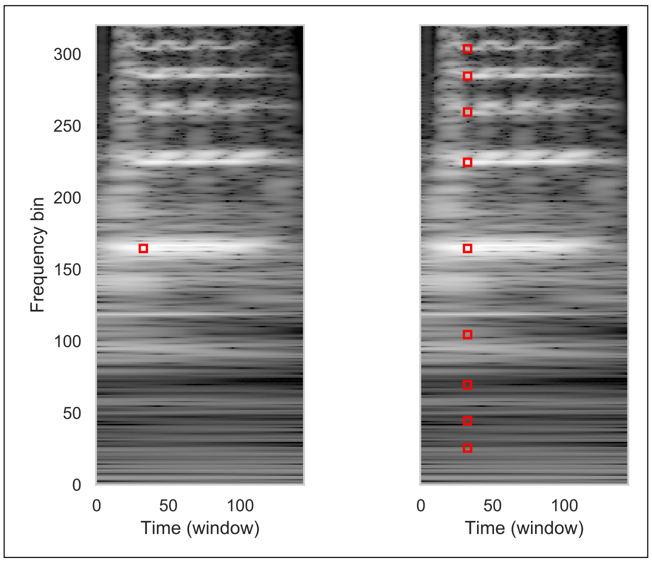

Figure 1. Comparison of receptive fields of standard CNN and HCNN on a CQT spectrogram.

time-frequency representation. In practice CNN kernels are usually small (previous works use usually a size of a semitone in the frequency axis [1, 3]) and therefore cannot process the whole harmonic structure in one layer. To make up for this, existing works also include a final "big" convolutional layer to "capture relationships between frequency content within a octave" [3]. This is only a partial solution because only one layer of the whole network can exploit the defining characteristic of a harmonic signal.

2. METHOD

Our proposed Harmonic Convolutional Neural Network (HCNN) overcomes the limitation outlined in the Introduction. The convolutional layer in our network is able to use all the relevant harmonic information in each layer as illustrated in Figure 1.

The architecture has the standard overall structure of processing an input through a series of convolutional layers. We use only convolutions with 1x1 stride and no pooling layers so that the input and output sizes stay the same throughout the whole network. The output of each layer is therefore a 3D matrix with channel, frequency and time axes of constant dimensions. Crucially before each convolutional layer we add an additional transformation of the

[1] For the source code please see the accompanying repository at https://github.com/kukas/music-transcription

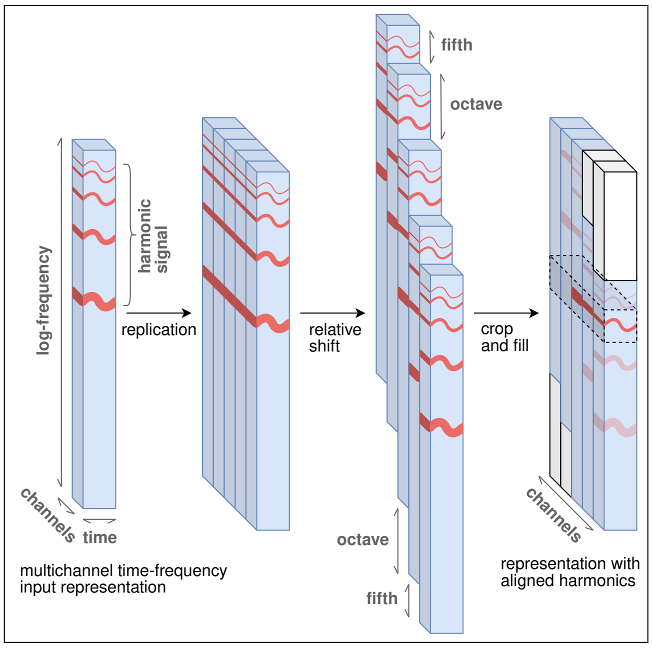

Figure 2. Diagram of the input transformation for convolution layers.

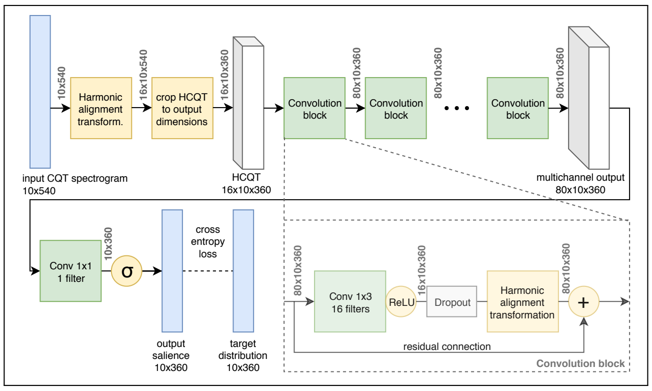

Figure 3. Overall HCNN architecture. The input spectrogram is 10 frames long in this figure. The actual number of frames however depends on the selected architecture (see Figure 4).

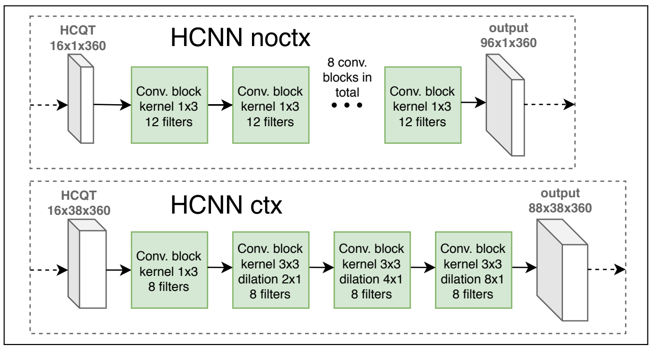
HCQT 16x1x360
HCNN noctx
output 96x1x360
Conv. block kernel 1x3 12 filters
Conv. block kernel 1x3 12 filters
8 conv. blocks in total
Conv. block kernel 1x3 12 filters
HCQT 16x38x360
HCNN ctx
output 88x38x360
Conv. block kernel 1x3 8 filters
Conv. block kernel 3x3 dilation 2x1 8 filters
Conv. block kernel 3x3 dilation 4x1 8 filters
Conv. block kernel 3x3 dilation 8x1 8 filters

Figure 4. Two tested architectures.

input (see Figure 2).

The transformation first creates copies of the input and stacks them in the channel axis, so that there are several identical copies of the input matrix. We then shift those copies relative to the original in the frequency axis so that the information about harmonically related peaks are positioned "above" [2] the original fundamental frequency. Since the input is a log-frequency spectrogram, the offsets are constant. For example in our case the shift of 60 bins corresponds to an octave (second harmonic frequency) and the shift of $60 + 35$ bins corresponds to an octave and a fifth (third harmonic frequency). In this way we can position as many harmonically related bins on top of each other as needed.

In addition to the shift to corresponding overtones, we also add copies shifted in the opposite direction. This allows the convolution to access corresponding fundamental frequency values when processing some of the overtones of a harmonic signal. We call those negative offsets "undertones".

This transformed input is then fed into the next convolutional layer. Since the convolutional filter has the access to all channels, it follows that it has access to the provided harmonically related information for every time-frequency position in the matrix.

2.1 Input representation

The input is a single CQT spectrogram with 5 bins per semitone and 540 bins in total (spanning the range from C1 to C9). Our hop size is $\frac{256}{44100} \approx 5.8\,\mathrm{ms}$. Note that CQT has logarithmically spaced frequency bins, which is an essential property for our method to work because it implies that distances between the harmonics are independent on the fundamental frequency of a signal. By applying the harmonic-alignment transformation explained in the previous section to this CQT input, we create a 3-dimensional array that closely resembles HCQT input representation used in [3] without the overhead of computing each CQT spectrogram separately. This allows us to cheaply increase the number of HCQT harmonics. We therefore create a HCQT representation by stacking 8 copies shifted by overtone offsets and 8 copies shifted by undertone offsets. Finally we crop the HCQT to match the output dimensions (360 bins, range from C1 to C6).

[2] "above" meant in the channel axis

2.2 Output representation

The output representation matches the cropped input frequency and time dimensions. In each timeframe the ground truth is represented as a gaussian with mean in the reference f_0 and standard deviation of 18 cents. Unvoiced frames are represented as zero vectors. Voicing output is obtained by thresholding the maximum value of each frame using a best performing threshold for the validation dataset. This representation is inspired by CREPE monopitch tracker [6], the std. dev. was finetuned for our purposes.

2.3 Architecture

The design of the rest of the network either draws from standard deep learning practices and from Bittner et al. [3]

Method	Parameters
SAL	—
BIT	406 253
BAS	307 199
HCNN noctx	24 961
HCNN ctx	19 513

Table 1. Comparison of the number of model parameters.

(see Figure 3). The input HCQT is processed using a stack of convolution blocks. The output is passed through a 1x1 convolution to shrink the number of channels and create a salience representation. The model is trained using cross entropy loss using Adam optimizer with learning rate 0.001 and 16 samples per batch.

A convolution block consists of a convolutional layer with 16 filters and ReLU activation, followed by a dropout layer (with dropout probability of 0.3) and the harmonic alignment transformation. We also add a residual connection [5] as it improves the model performance with no significant additional overhead.

We test two architectures, HCNN noctx and HCNN ctx. HCNN noctx consists of 8 convolution blocks that contain convolutional layers with kernel size 3x1 and 12 channels. In other words the convolutions operate only in the frequency dimension and thus the estimation is based only on one frame of the input spectrogram ($\approx$ 5.8 ms of audio). The harmonic aligment transformation for this architecture includes three undertones and four overtones (channels shifted by -24 -19, -12, 0, +12, +19, +24 and +27.8 semitones).

The HCNN ctx architecture consists of 4 convolution blocks. The first block with kernel size 3x1 and the rest with sizes 3x3. Additionaly we set an increasing dilation rate in the last three convolution blocks (along the lines of the WaveNet architecture [13]). In this way we achieve a receptive field of $\approx$ 162.4 ms using only three layers that operate also in the time dimension. Note that this resulting receptive field is comparable to Bittner et al. [3]. The harmonic aligment transformation for this architecture includes five undertones and five overtones.

3. EXPERIMENTS

3.1 Datasets

For training and validation, we use MedleyDB [2]. We use the same dataset split as in [1,3]. This has the advantage of being able to directly compare the results of the methods on the testing split.

Besides the MedleyDB testing set, for testing we also include ADC04 dataset, MIREX05 training set [3], Orchset [4], a subset of MDB-melody-synth [11] and a subset of WJAZZD [9]. [4]

[3] We downloaded ADC04 and MIREX05 datasets from `https://labrosa.ee.columbia.edu/projects/melody/`

[4] For the complete list of selected testing tracks please see the repository at `https://github.com/kukas/music-transcription`.

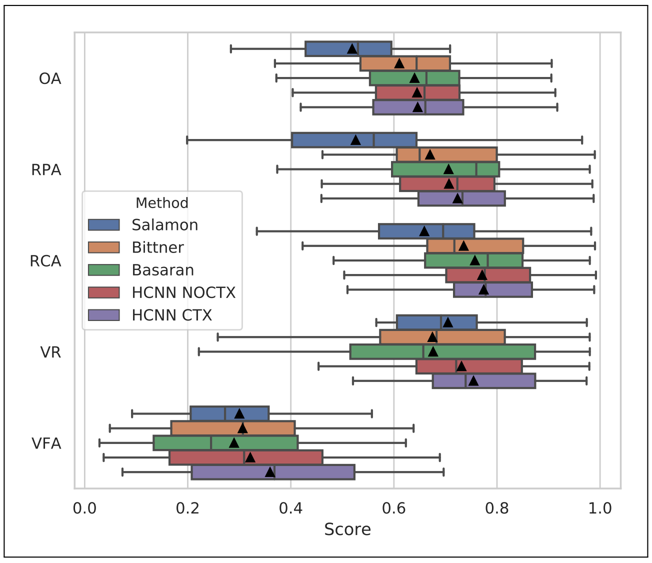

Figure 5. Evaluation metrics on MedleyDB testing split for SAL, BIT, BAS, HCNN noctx and HCNN ctx.

Method	ADC04	MDB-m-s test	MIREX05 train.	MDB test	ORCH-SET	WJazzD test
SAL	0.714	0.527	0.715	0.519	0.235	0.667
BIT	0.716	0.633	0.702	0.611	0.407	0.692
BAS	0.669	**0.689**	**0.734**	0.640	0.483	0.700
HCNN noctx	0.735	0.636	0.728	0.645	0.457	**0.714**
HCNN ctx	**0.746**	0.637	0.704	**0.646**	**0.525**	0.711

Table 2. Overall Accuracy of the selected methods. Highlighted values are the highest across the dataset.

3.2 Methodology

For evaluation, we use the melody extraction evaluation functions from the library `mir_eval` [5]. These include the standard metrics used in the context of Melody Extraction. We compare the output of our models "HCNN noctx" (HCNN with minimal audio context) and "HCNN ctx" (HCNN with broader audio context) with state-of-the-art baselines: "SAL" [12], "BIT" [3] and "BAS" [1]. In case of BIT and BAS, we ran the algorithms using the source code obtained from the links provided in the papers keeping default parameters. For the SAL algorithm we used the implementation in Essentia library [6]. All the output melody estimates are included in the repository.

3.3 Results

We present a detailed comparison of the results of the selected methods on MedleyDB testing split on Figure 5 and a overview of Overall Accuracy on six different datasets in Table 2. Our methods outperform all selected baselines on four out of six testing datasets. Compared to the most similar method of Bittner et al. [3] we achieve better results on all datasets on the Overall Accuracy (OA) metric. Basaran et al. [1], who additionally use a recurrent layer on

[5] `https://github.com/craffel/mir_eval`

[6] `https://essentia.upf.edu`

top of the convolutional stack, therefore giving the network an advantage over the HCNN and Bittner by allowing it to learn also how melody works in time, achieved better results on two datasets. Compared to the methods based on deep learning (Bittner and Basaran), our model uses between 10 and 20 times less parameters (see Table 1).

Qualitatively we see an expected difference between the outputs of HCNN noctx, HCNN ctx and BAS. The results of HCNN noctx are noisier compared to HCNN ctx and BAS, since one audio input frame is only $5.8\,\mathrm{ms}$ long. It is impossible for this method to take into account any musical context even within one continuous note. For pieces with higher polyphony, the output of HCNN noctx therefore jumps between possible melody f_0 candidates.

The results of HCNN ctx and Bittner are highly correlated, from a closer qualitative inspection we conclude that the performance difference can be attributed to a better generalization on instrument classes underrepresented in the training dataset and better learning of the instrument priorities.

4. CONCLUSION

In this paper we presented a novel and general way to adapt ordinary CNN-based networks for processing harmonic audio signals. We showed that using HCNN layers is effective for the Melody Extraction task and yields state-of-the-art results while dramatically reducing the needed number of parameters of the model. Based on these results we believe that the architecture has a potential in other related tasks such as multi-f_0 tracking.

Possible improvements include data augmentation (see [8] for an interesting new data augmentation technique designed for audio data) and temporal smoothing of the output using HMM, LSTM or other techniques like in Basaran et al. [1]. Other potential improvements can be done by altering the input representation or the model architecture.

5. ACKNOWLEDGMENTS

We would like to thank the Jazzomat Research Project for kindly providing the WJazzD testing data.

6. REFERENCES

[1] Dogac Basaran, Slim Essid, and Geoffroy Peeters. Main Melody Estimation with Source-Filter NMF and CRNN. *Proceedings of the 19th International Society for Music Information Retrieval Conference*, pages 82–89, 2018.

[2] Rachel Bittner, Justin Salamon, Mike Tierney, Matthias Mauch, Chris Cannam, and Juan Bello. MedleyDB: A multitrack dataset for annotation - intensive mir research. *International Society for Music Information Retrieval Conference*, 2014.

[3] Rachel M. Bittner, Brian Mcfee, Justin Salamon, Peter Li, and Juan P. Bello. Deep Salience Representations for F0 Estimation in Polyphonic Music. *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR 2017)*, pages 63–70, 2017.

[4] Juan J. Bosch, Ricard Marxer, and Emilia Gómez. Evaluation and combination of pitch estimation methods for melody extraction in symphonic classical music. *Journal of New Music Research*, 45(2):101–117, 2016.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-December:770–778, 2016.

[6] Jong Wook Kim, Justin Salamon, Peter Li, and Juan Pablo Bello. CREPE: A Convolutional Representation for Pitch Estimation. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 161–165, 2018.

[7] Sangeun Kum, Changheun Oh, and Juhan Nam. Melody Extraction on Vocal Segments Using Multi-Column Deep Neural Networks. *Proceedings of the 17th International Society for Music Information Retrieval Conference, (ISMIR)*, pages 819–825, 2016.

[8] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition. *Interspeech 2019*, 2019.

[9] Martin Pfleiderer, Klaus Frieler, Jakob Abeßer, Wolf-Georg Zaddach, and Benjamin Burkhart, editors. *Inside the Jazzomat*. Schott Campus, 2017.

[10] Francois François Rigaud and Mathieu Radenen. Singing Voice Melody Transcription using Deep Neural Networks. *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 737–743, 2016.

[11] Justin Salamon, Rachel M. Bittner, Jordi Bonada, Juan J. Bosch, Emilia Gomez, and Juan Pablo Bello. An Analysis/Synthesis Framework for Automatic F0 Annotation of Multitrack Datasets. *18th International Society for Music Information Retrieval Conference*, pages 71–78, 2017.

[12] Justin Salamon and Emilia Gomez. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Transactions on Audio, Speech and Language Processing*, 20(6):1759–1770, 2012.

[13] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. WaveNet: A Generative Model for Raw Audio. pages 1–15, 2016.