MAIN MELODY EXTRACTION WITH SOURCE-FILTER NMF AND CRNN

Dogac Basaran[1] **Slim Essid**[2] **Geoffroy Peeters**[1]

[1] CNRS, Ircam Lab, Sorbonne Université, Ministère de la Culture, F-75004 Paris, France
[2] LTCI, Télécom ParisTech, Université Paris Saclay, Paris, France

dogac.basaran@ircam.fr

ABSTRACT

Estimating the main melody of a polyphonic audio recording remains a challenging task. We approach the task from a classification perspective and adopt a convolutional recurrent neural network (CRNN) architecture that relies on a particular form of pretraining by source-filter nonnegative matrix factorisation (NMF). The source-filter NMF decomposition is chosen for its ability to capture the pitch and timbre content of the leading voice/instrument, providing a better initial pitch salience than standard time-frequency representations. Starting from such a musically motivated representation, we propose to further enhance the NMF-based salience representations with CNN layers, then to model the temporal structure by an RNN network and to estimate the dominant melody with a final classification layer. The results show that such a system achieves state-of-the-art performance on the MedleyDB dataset without any augmentation methods or large training sets.

1. INTRODUCTION

Automatic dominant melody estimation (AME) is a popular and rather challenging task in Music Information Retrieval (MIR). In general, AME can be defined as the estimation of fundamental frequencies that represent the pitch values of the dominant melody [24]. The source of the dominant melody could be a leading singing voice or an instrument. The difficulty is that there is usually a polyphonic accompaniment to the lead vocal/instrument, and that this accompaniment follows the melody rhythmically and harmonically, in the sense that chord progressions will naturally contain the dominant F_0 and/or its harmonics. As a consequence, it is not trivial to obtain a representation that discriminates the main melody from the background music. Hence, one of the main research directions in AME remains finding a salience representation that enhances the fundamental frequency of the dominant melody against the possibly polyphonic background.

One of the most popular and rather simple salience representations is the Harmonic Sum Spectrum (HSS) [18] that consists of mapping the energy among harmonically related F_0s. This has been used effectively in a popular melody extraction algorithm, jbcorso-called *Melodia* [23]. Durrieu et. al. [11, 12] proposed a salience function where the dominant melody (singing voice or instrument) is modeled with a Source-Filter Nonnegative Matrix Factorization (SF-NMF). This method was later combined with HSS in [7] in order to obtain an enhanced salience representation. There also exist other methods that utilize a simple time-frequency representation, e.g., the Short Time Fourier Transform (STFT) or Constant Q-Transform (CQT), as a low-level representation of salience [13, 25].

Recently, Bittner et. al. [6] proposed a Convolutional Neural Network (CNN) system to learn salience representations based on harmonic CQT. The rationale for this approach is to learn harmonic relationships implicitly and to obtain a salience representation similar to (or better than) HSS.

Salience-based melody estimation methods usually use pitch tracking methods on top of salience representations to exploit the temporal relationships between dominant F_0s. In [12], a Hidden Markov model (HMM) was adopted where the states represent the bins of the source activations, i.e. F_0s. Then a threshold-based voicing estimation (melody/non-melody estimation) was applied. Another very popular pitch tracking method was proposed by Salamon et. al. [23] where the algorithm creates and characterizes pitch contours on top of HSS. Characteristics of these contours have proven very effective in voicing estimation [7, 23].

Recently, Deep Neural Networks (DNNs) have become very popular in MIR applications such as sound event detection [2, 4] and chord estimation [20]. The ability of DNNs to approximate any function with linear weights and non-linear activations, given enough data, makes such systems attractive for MIR tasks. That said, comparatively few attempts have been made to estimate dominant melody using neural networks. In [19, 22], bidirectional Long Short-Term Memory (LSTM) [15], a special kind of Recurrent Neural Network (RNN), are used for singing voice separation. Such networks are mostly used in modeling the

temporal information in time sequences. Recently, in [3], a hierarchical CNN structure similar to a stacked denoising autoencoder (SDA) [26] is used to learn a mapping between an STFT representation and a transcription similar to a piano roll. A tutorial on deep learning techniques for MIR tasks can be found in [9].

Although most of these DNNs perform end-to-end training, it has proven effective to use a more structured input data, such as harmonic-CQT [6]. Recently, [4] achieved state-of-the-art results in sound classification by using NMF activations as input as a form of pretraining.

Contributions. Inspired by these works, we propose a Convolutional-Recurrent Neural Network (CRNN) model whose pretraining is based on the SF-NMF model proposed in [12]. We show that with NMF-based pretraining, we can achieve state-of-the-art results without requiring large training datasets or data augmentation methods, and using relatively simpler networks in terms of training parameters. Our results clearly demonstrate the usefulness of a good input salience representation to the network, suggesting that performance would climb even higher if the SF-NMF model were improved.

Our results are obtained on MedleyDB [5], which is a challenging dataset due to inclusion of singing voice and instrument melodies in a diverse set of music genres.

The rest of the paper is organized as follows: the proposed CRNN system and pretraining with SF-NMF are detailed in Section 2. Section 3 discusses the dominant melody estimation results obtained on the MedleyDB dataset, and also gives an analysis of SF-NMF-based salience and the comparison between different CRNN variants. Finally, some conclusions and future directions are given in Section 4.

2. SYSTEM OVERVIEW

The block diagram of the CRNN system we propose is given in Figure 1. In the first stage (Pretraining), we estimate an initial salience representation using the SF-NMF model. Then this salience is fed into a CNN (CNN stage), where the salience representation is further enhanced by learning local features. The CNN output activations are then fed into an RNN to exploit the long-term relationships between fundamental frequencies (RNN stage). Then in the final Classification stage, we classify the representations as melody/non-melody and give an estimate for F_0 at each time-frame where each class represents a semitone fundamental frequency. Note that the same procedure is applied in both the training and testing of the system.

In the design of the CRNN system, we are inspired by a similar CRNN proposed in [20] for chord recognition, where the network is interpreted as an encoder-decoder scheme. In the CRNN structure we propose, the CNN and RNN stages can also be treated together as an encoding stage (input sequence to mid-level salience representation) where the output is an enhanced salience representation that captures both spatial and temporal features. Then the classification stage acts as a decoding stage (mid-level representation to output sequence) where the salience is mapped into a frame-based note representation.

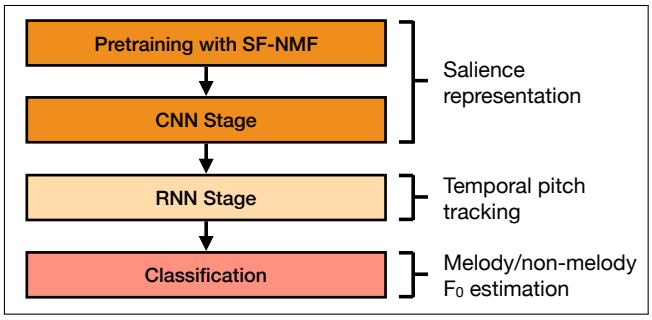

Figure 1: Block diagram of the proposed CRNN system with pretraining

2.1 Pretraining with SF-NMF

In [12], the dominant melody (voice/instrument) is modeled using a source-filter model. Assuming the mixing of the dominant melody and the accompaniment (background) is instantaneous, the source, filter and accompaniment parts are modeled with the SF-NMF model as follows:

$$\begin{aligned}\mathbf{V} \approx \hat{\mathbf{V}} &= \mathbf{V}^{F_0} \odot \mathbf{V}^{\Phi} + \mathbf{V}^{B} \\ &= \mathbf{W}^{F_0}\mathbf{H}^{F_0} \odot \mathbf{W}^{\Phi}\mathbf{H}^{\Phi} + \mathbf{W}^{B}\mathbf{H}^{B} \quad (1) \\ &= \mathbf{W}^{F_0}\mathbf{H}^{F_0} \odot \mathbf{W}^{\Gamma}\mathbf{H}^{\Gamma}\mathbf{H}^{\Phi} + \mathbf{W}^{B}\mathbf{H}^{B} \quad (2)\end{aligned}$$

where $\mathbf{V}$ represents the power spectrogram of the signal, i.e., $\mathbf{V} = |\mathbf{X}|^2$ (where $\mathbf{X}$ is the STFT of the audio signal to be analyzed); F_0, Φ and B represents the source, filter and background respectively; $\mathbf{W}$ and $\mathbf{H}$ represent the basis and activation matrices; and $\odot$ denotes the Hadamard product. The filter basis $\mathbf{W}^{\Phi}$ is further modeled with yet another NMF representation, as in [11]: $\mathbf{W}^{\Phi} = \mathbf{W}^{\Gamma}\mathbf{H}^{\Gamma}$.

In this model, the source, $\mathbf{V}^{F_0} = \mathbf{W}^{F_0}\mathbf{H}^{F_0}$, is assumed to have a harmonic structure. To ensure such a structure, the basis $\mathbf{W}^{F_0}$ is pre-constructed (not estimated) such that each column represents the harmonic structure for one F_0. Represented F_0s start from a minimum frequency, i.e., $F_0 = 55Hz$, and they are logarithmically spaced, i.e., the ratio between consecutive F_0 values would be $2^{(1/60)}$ for a resolution of 5 bins per semitone. Such a construction enforces the corresponding row in the activation matrix $\mathbf{H}^{F_0}$ to represent the activation of that specific F_0, similar to a saliency representation. That is the rationale behind using $\mathbf{H}^{F_0}$ as a saliency representation as in [7, 11, 12].

The main assumption with the filter, $\mathbf{V}^{\Phi}$, is to have a smooth structure. One way to ensure such smoothness is to construct a basis $\mathbf{W}^{\Phi}$ from smooth filters in advance, similar to enforcing harmonic structure in the source $\mathbf{V}^{F_0}$. However it is not possible to directly construct $\mathbf{W}^{\Phi}$ with smooth basis filter structures since it depends on the dominant melody. In [11], it is proposed to represent $\mathbf{W}^{\Phi}$ with another NMF model, $\mathbf{W}^{\Gamma}\mathbf{H}^{\Gamma}$, where the columns of $\mathbf{W}^{\Gamma}$ are constructed (not estimated) as simple and smooth band pass filters that are linearly spaced and overlapping. This structure forces $\mathbf{W}^{\Phi}$ to be smooth, thus ensuring that $\mathbf{V}^{\Phi}$ will be smooth as expected.

The accompaniment/background, $\mathbf{V}^B = \mathbf{W}^B\mathbf{H}^B$, is also represented with a standard NMF model where there are no constraints on the basis such as smoothness or being harmonic. In summary, the source basis $\mathbf{W}^{F_0}$ and smooth filter basis $\mathbf{W}^{\Gamma}$ are pre-constructed and the rest of the parameters $\mathbf{H}^{F_0}$, $\mathbf{H}^{\Gamma}$,$\mathbf{H}^{\Phi}$,$\mathbf{W}^B$ and $\mathbf{H}^B$ are estimated using the standard alternating scheme and heuristic multiplicative updates.

In this work, for the SF-NMF model, we follow the parametrization given in [7] where the minimum and maximum frequencies represented in $\mathbf{H}^{F_0}$ are chosen as $55Hz$ and $1760Hz$ respectively. We choose the resolution of the F_0s as 5 bins per semitone which results in 60 bins per octave (bpo) and 301 bins in total per frame.

Note that due to the logarithmic spacing of the F_0s where the consecutive frequencies have a ratio of $2^{1/60}$, one can tune the represented F_0s with proper choice of the minimum frequency $F_{0,min}$. As an example, if $F_{0,min} = 55Hz$, the notes will be tuned to $A4 = 440Hz$ whereas if $F_{0,min} = 55.25Hz$, they will be tuned to $A4 = 442Hz$. This choice of tuning might depend on the target dataset. Here, we choose the tuning $A4 = 440Hz$ assuming that such tuning is more widely used. It is important to mention that this construction of F_0s in $\mathbf{W}^{F_0}$ cannot be generalized to all music genres, e.g., traditional Turkish music with makams. Hence the methods based on SF-NMF, as well as the proposed scheme, are limited in that sense.

Although we aim to classify the fundamental frequencies at semitone resolution, we initially choose a higher resolution for the F_0s in $\mathbf{W}^{F_0}$. In practice, it is highly probable that a dominant voice or instrument will be slightly out-of-tune, and hence will not fit any of the represented F_0s. In such cases, a high resolution representation of F_0s might better describe these out-of-tune notes.

2.2 CNN stage

In order to enhance the $\mathbf{H}^{F_0}$-salience, we propose two different CNN architectures, which we denote as CNN1 and CNN2. In CNN architecture 1 (CNN1), we first decrease the F_0 resolution to semitones, then we train CNN layers to learn local structures, i.e., the confusions between semitones. In the second approach (CNN2), we follow the network proposed in [6]. Here, the network learns the features in the original resolution and within a semitone interval with one additional layer that learns the octave patterns.

Note that since each CNN architecture only applies 2D linear filters and non-linear activations, the input structure is not lost through the layers of the network. This provides an advantage of interpretable hidden layer activations and leads to a new form of salience as output where each row still represents the activation of a fundamental frequency.

In both architectures, rectified linear units (ReLus) are used as non-linear activations and are applied to each CNN layer output. Batch normalization is applied before each intermediate CNN layer input, as it has proven effective in the convergence of the network by reducing the internal covariance shift [16]. The columns of $\mathbf{H}^{F_0}$ are normalized with l_1 norm before being fed into the CNN network. Such a normalization is possible since the task at hand is the estimation of the melody; that is, only the position of the fundamental frequency is needed, not the exact energy.

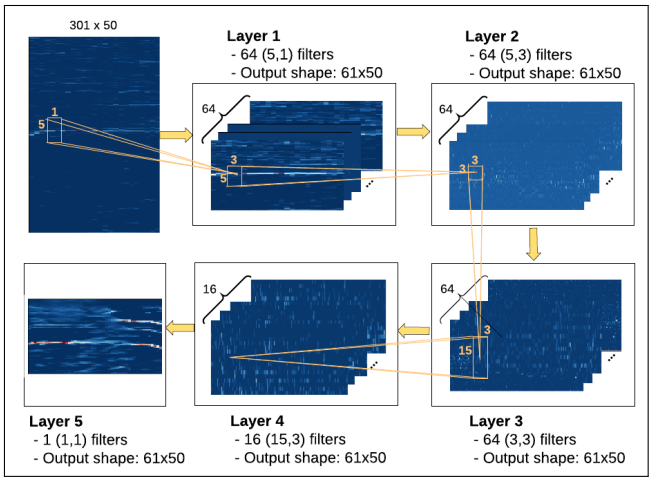

Figure 2: CNN Architecture 1 (CNN1).

2.2.1 *CNN Architecture 1 (CNN1)*

There are 5 layers in the CNN1 architecture. The first layer gathers the energy around each semitone by applying focused filters centered around each semitone frequency. In this layer, there are 64 (5x1) filters each with a stride (5,1). This way, not only is the energy focused on the semitones, but also the frequency resolution is decreased to the semitone scale from 5 bins per semitone (time resolution remains the same). The rationale behind the first layer is two-fold: First, the number of parameters is severely decreased by lowering the frequency resolution, i.e., it takes 5 times less filter parameters in order to learn features. Second, out-of-tune notes would already be represented in the vicinity of the corresponding semitone in the $\mathbf{H}^{F_0}$ representation. Focused filters on semitones would gather the energy on the semitone that is a way of retuning the melody on the represented semitone fundamental frequencies.

In the following layers, zero padding is applied to convolutions to keep the dimensions unchanged. The second layer has 64 (5 x 3) filters that cover ± 2 semitone interval and roughly 30ms in time. Then the third layer has 64 (3 x 3) filters that cover ± 1 semitone and 30ms in time. The fourth layer has 16 (15 x 3) filters to learn note confusions in one octave. Filters cover ± 7 semitone interval and again 30ms in time. Then enhanced salience representation is obtained as the output of the final CNN layer that has only one (1x1) filter as in [6] but with a rectified linear unit instead of a sigmoid. The overall structure of CNN architecture 1 is shown in Figure 2.

2.2.2 *CNN Architecture 2 (CNN2)*

CNN2 is based on the network proposed in [6]. In this network, the resolution of the input remains the same throughout the layers of the CNN, i.e., no pooling is applied. Note that the input to CNN2 is $\mathbf{H}^{F_0}$; therefore, the first layer of the network contains only a single channel instead of six.

As mentioned before, the overall system targets semitone resolution for the output fundamental frequencies. This requires a reduction in resolution somewhere in the system. In this architecture, we left the dimensionality reduction to the final classification layer.

2.3 RNN stage

Recurrent neural networks are mostly used in MIR and audio analysis tasks to model the dynamics of the observations, typically for chord recognition [20] and speech recognition [14]. Here, we use a single bidirectional Gated Recurrent Unit (BiGRU) layer to capture temporal relationships between F_0s. A GRU is a special kind of RNN [8] where the units are able to model long-term temporal relationships whilst using a gate structure. It has the advantage of not suffering from the vanishing gradient problem of standard RNN and has proven to be easier to train compared to the LSTM alternative.

The number of units in a BiGRU layer should be chosen higher or equal to the output dimension of the preceding CNN network. In the BiGRU structure, actually two GRU layers are trained with the same input but in reverse directions to model the F_0 relationships from both directions. Later, the two layers are merged to have a single output.

2.4 Classification

The final layer of the system is a classifier where one class represents the non-melody and the rest of the 61 classes represent semitone fundamental frequencies between A1 and A6 (inclusive). The multiclass classification output is obtained with a single dense layer and softmax activations.

The overall system is trained minimizing the cross entropy loss between the softmax activations and true probabilities. A frame is classified as a non-melody frame only if the probability of non-melody class is higher than the rest. Regardless of this decision, F_0 is estimated for each frame by simply picking the most probable F_0 class among the 61 note classes. Note that even if the non-melody class has the highest probability, the second-highest probability gives a good estimation of the pitch.

An example output of the classification layer that is obtained from a CNN1 + RNN + Classification architecture is shown in Figure 3. In this example, $\mathbf{H}^{F_0}$ input (top-left) gives a very good initial salience. Then the CNN1 output activations (top-right) further enhance the dominant part against the harmonic background. It is observed that the dominant F_0 classes mostly have the highest probabilities against the rest of the class probabilities (bottom-left).

3. EXPERIMENTS

In this section, we evaluate the proposed NMF-based CRNN system using the MedleyDB dataset [5]. For the annotations, we use the "Melody2" definition in MedleyDB that is the F_0 of the dominant melody at each time step, drawn from multiple sources. With this definition of melody, it is possible to have separate instruments or

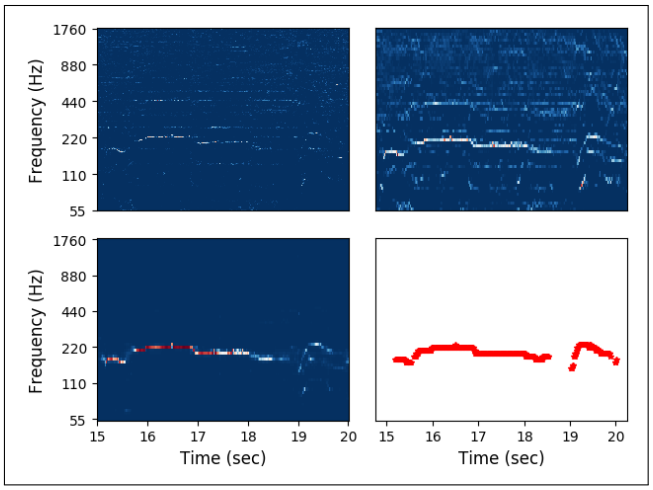

Figure 3: (Top-left) $\mathbf{H}^{F_0}$ representation of a small audio excerpt as input to CRNN, (Top-right) CNN1 activations, (Bottom-left) Classifier activations of CRNN, (Bottom right) Ground-truth annotations.

voices as the source of dominant melody throughout a single song. Among 108 annotated songs in the dataset, 48 songs have predominant instrumental melody, 30 songs have predominant vocal melody and 30 songs have both predominant instrument and vocal melodies.

We randomly split the MedleyDB set into train, validation and test sets such that the tracks from the same artist do not belong to different sets following the artist conditional random splitting as in [6,7]. There are 27 full-length tracks in the test set, 67 full-length tracks in the training set and 14 full-length tracks in the validation set. Note that we used the same test split with [6] in the MedleyDB in the rest of the experiments to be able compare the results.

We use the five standard evaluation metrics given in [24], namely: Raw Pitch Accuracy (RPA), Raw Chroma Accuracy (RCA), Overall Accuracy (OA), Voicing False Alarm (VFA) and Voicing Recall (VR). All the codes are written in Python and available online [1]. CQT implementation is based on the *librosa* python package [21].

3.1 Network training

We trained three different networks with the following combinations of the architectures given in Section 2:

CRNN-1: CNN1 + 1 layer BiGRU (128 Units) + Classification layer;
CRNN-2: CNN2 + 1 layer BiGRU (160 Units) + Classification layer;
C-NN: CNN2 + Classification layer.

We further denote the network variants by prepending a label indicating the input to the network: "SF" for $\mathbf{H}^{F_0}$ input and "CQT" for CQT input. Note that the CQT parameters are chosen such that the representation of a signal via $\mathbf{H}^{F_0}$ or CQT would have the same dimensions [2].

[1] github.com/dogacbasaran/ismir2018_dominant_melody_estimation
[2] CQT parameters: Minimum F_0=55Hz, # of octaves = 5, bpo = 60

	CRNN-1	CRNN-2	Baseline
# of Param.	307,199	854,319	406,253

Table 1: The number of trainable parameters for CRNN-1, CRNN-2 and the baseline CNN network [6]

In the proposed CRNN structure, the purpose of the CNN stage is to learn local features, whereas the purpose of the RNN stage is to account for long term temporal relationships. This requires selecting relatively small patch lengths for the CNN layers but longer patch lengths for the RNN layer. For this purpose, we used different patch lengths for the CNN and RNN parts while jointly training them.

In all the models, the CNN layers are trained on either 0.29-second (25-frame) or 0.58-second (50-frame) patches and the RNN layer is trained on 5.8-second (500-frame) patches. The training is performed using mini-batches of 16 patches per batch. We use the ADAM optimizer [17], and reduce the learning rate if there is no improvement in validation loss after 20 epochs. The early stopping strategy is used if the validation loss is not decreased after 20 epochs. The maximum possible number of epochs is set to 200. All models were implemented with Keras 2.0 [10] and Tensorflow 1.0 [1] and tested using NVIDIA-Tesla K80 GPUs. The number of parameters for each network model is given in Table 1.

Note that, in the training, we do not benefit from any data augmentation method or from other larger datasets.

3.2 Results

We compare the outputs of all three models to a CNN-based melody tracking system [6], considered as a baseline, which proved to significantly outperform the previous state-of-the-art methods in [7, 23]. The evaluation results of [6] are available online. [3] By choosing the same test split from the MedleyDB, we are able to compare these published results to ours without any re-evaluation. The evaluation results for all network variants (SF-CRNN-1, SF-CRNN-2, CQT-CRNN-2, SF-C-NN) and for the baseline are given in Figure 4. We use McNemar's test on the classification results and provide p-values as a measure of significance whenever relevant [4].

CQT vs. $\mathbf{H}^{F_0}$ as salience

We explore the usefulness of pretrained input by comparing the evaluation results of the CRNN-2 model when the input is CQT or $\mathbf{H}^{F_0}$—i.e., comparing CQT-CRNN-2 and SF-CRNN-2. The results show that CRNN-2 model performs significantly better in OA (p=0.0015) and RCA (p=0.0003) scores when the input to the network is $\mathbf{H}^{F_0}$. On average, results for SF-CRNN-2 are 6, 9 and 7 percentage points higher for OA, RPA and RCA, respectively.

The reason the CRNN-2 model performs better with pretrained input is that $\mathbf{H}^{F_0}$ provides a better initial salience representation than the CQT. Ideally, a salience representation of melody should be discriminative for each target fundamental frequency against the polyphonic background music. We can analyze both $\mathbf{H}^{F_0}$ and CQT representations to see how well they fit this definition of "ideal" salience by performing a simple peak-picking strategy as in [6]. Specifically, the frequency with maximum amplitude/salience for each time frame point is chosen as the estimate of the fundamental frequency. We can compute the RPA and RCA scores using those estimates to see their performances as salience. The results obtained on the full MedleyDB dataset are given in Table 2. It can be seen that $\mathbf{H}^{F_0}$ performs nearly twice as well as the CQT representation in both RPA and RCA scores, showing that $\mathbf{H}^{F_0}$ provides a better initial salience to the CRNN networks.

[3] github.com/rabitt/ismir2017-deepsalience

[4] Mcnemar test is based on *statsmodel* package in python.

	$\mathbf{H}^{F_0}$	CQT
RPA	0.538 ± 0.141	0.210 ± 0.16
RCA	0.648 ± 0.127	0.411 ± 0.15

Table 2: The comparison of RPA and RCA scores for $\mathbf{H}^{F_0}$ feature and CQT feature by simple peak-picking method.

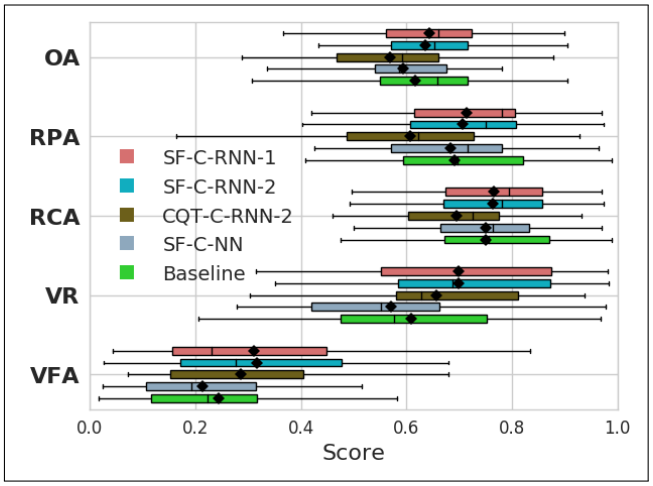

Figure 4: Evaluation metrics for SF-CRNN-1, SF-CRNN-2, CQT-CRNN-2, SF-C-NN and the baseline [6].

SF-CRNN-2 model vs. Baseline CNN Network

The SF-CRNN-2 model uses the CNN-2 architecture in the CNN stage, the same CNN as the baseline. When we compare the evaluation results given in Figure 4, we observe that the SF-CRNN-2 model outperforms the baseline in the RPA (p = 0.0015) and VR (p=0.052) scores. The model has slightly higher OA and RCA scores on average than the baseline. On the other hand, SF-CRNN-2 has a higher number of network parameters $(854,319)$ than the baseline CNN $(406,253)$. This is due to the additional RNN layer that exists in SF-CRNN-2.

Comparison between variants SF-CRNN-1, SF-CRNN-2 and SF-C-NN

On average, SF-CRNN-1 performs slightly better than all other models in all metrics aside from VFA. Comparing SF-CRNN-1 and SF-CRNN-2, we observe that a similar or

	OA	RPA	RCA	VR	VFA
SF-CRNN-1	0.444	0.595	0.677	0.556	0.423
Baseline	0.580	0.756	0.725	0.590	0.219

Table 3: Evaluation results for the track "MatthewEntwistle_TheFlaxenField" where the worst OA performance occurs against the baseline [6].

higher performance can be achieved by the low resolution CNN1 architecture and with far fewer training parameters (see Table 1). VR rates for SF-CRNN-1 and SF-CRNN-2 are significantly higher than the SF-C-NN; however, VFA rates are higher as well. This behavior could be due to the activations of the RNN layer that should force some sort of temporal smoothing on the salience representation.

On the other hand, the significantly better OA, RPA and RCA scores of SF-CRNN-2 relative to SF-C-NN suggest that the temporal tracking with RNN effectively improves the performance of the melody estimation.

Comparing the best performing network variant SF-CRNN-1 to the baseline, we observe that it outperforms the baseline on the OA (p=0.052), RPA (p=0.0003) and VR (p=0.0015) scores, and achieve those results with a less complex network in terms of network parameters (see Table 1). A track-level comparison by computing the overall accuracy differences for each track shows that SF-CRNN-1 performs better on 19 tracks out of 27.

The worst OA of SF-CRNN-1 occurs against the baseline with the "MatthewEntwistle_TheFlaxenField" track where the dominant melody consists only of instruments including Piano. The evaluation results for this track are given in Table 3. It is observed that both SF-CRNN-1 and baseline have relatively high VFA; however, the effect of this is minimal since the track mostly contains voiced frames. On the other hand, the OA score would be highly affected by the combination of high RPA and VR scores. For this track, although the baseline and SF-CRNN-1 have comparable VR rates, the RPA score of the baseline is better, which explains the difference in OA performance.

Singing voice vs. Instrument

Among the test set in MedleyDB, 16 tracks contain only instrumental dominant melody, 3 tracks contain only dominant singing voice melody and 8 tracks contain both [5]. Evaluation results in Table 4 show that SF-CRNN-1 performs better for singing voice melodies than instrument melodies. SF-CRNN-1 outperforms the baseline in overall accuracy for singing voice melodies and instrument melodies.

	SF-CRNN-1		Baseline	
	S.V.	Ins.	S.V.	Ins.
OA	0.638	0.466	0.598	0.424
RPA	0.791	0.647	0.784	0.619
RCA	0.804	0.726	0.823	0.717

Table 4: OA, RPA and RCA scores for singing voice (S.V.) main melody and Instrument (Ins.) main melody for SF-CRNN-1 and baseline.

4. CONCLUSIONS AND FUTURE WORK

In this work, we have introduced a novel audio-based dominant melody estimation architecture using source-filter NMF as pretraining for a new variant of deep network for this task, namely a CNN-BiGRU scheme. We have shown that the proposed system achieves state-of-the-art performance on standard evaluation metrics, even significantly improving on it while maintaining a lower system complexity.

Analysis of $\mathbf{H}^{F_0}$ as a salience representation shows that it provides a good initial salience in general with high RPA and RCA, even when performing melody estimation using frame-based salience peak-picking. The evaluation results clearly show the usefulness of SF-NMF-based pretraining in many aspects. We observe that when provided with a good initial salience input to the CRNN structure, the system performs considerably better without requiring any augmentation or additional training data. This encourages the idea of improving the pretraining part to obtain even more discriminative salience representations which will surely increase the melody estimation performance. For such improvements, SF-NMF is a good candidate since many other variants with various constraints such as smoothness or sparsity exist in the literature.

We observe that in the proposed CRNN structure, the CNN stage helps to improve the quality of the salience representation against $\mathbf{H}^{F_0}$. In addition, exploiting temporal information with the RNN significantly improves OA, RPA, RCA and VR. These two stages act similarly to an encoder scheme and the classification layer acts as the decoder. Therefore one can interpret the proposed CRNN as an encoder-decoder network where the encoder is used to obtain an enhanced salience representation and the decoder produces a frame-based transcription.

From a melody classification viewpoint, the MedleyDB dataset is quite challenging due to its diverse range of instrumentation and music genres. Also, there is an imbalance between the note classes and the non-melody class in the dataset. The CRNN network has proven effective in handling such imbalance when pretrained with an SF-NMF model.

A clear future direction to pursue is training the SF-NMF and CRNN jointly, learning the $\mathbf{H}^{F_0}$ representation while minimizing the classification error.

5. ACKNOWLEDGEMENT

This project is partly funded by the *DigThatLick* project. We'd like to thank Rachel Bittner, Dr. Umut Simsekli and Dr. Jordan Smith for their valuable technical support.

[5] The ratio of the dominant singing voice melody frames and the dominant instrumental melody frames among all voiced frames is 0.238 and 0.762, respectively.

6. REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Sharath Adavanne, Konstantinos Drossos, Emre Çakir, and Tuomas Virtanen. Stacked convolutional and recurrent neural networks for bird audio detection. In *Signal Processing Conference (EUSIPCO), 2017 25th European*, pages 1729–1733. IEEE, 2017.

[3] Stefan Balke, Christian Dittmar, Jakob Abeßer, and Meinard Müller. Data-driven solo voice enhancement for jazz music retrieval. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 196–200. IEEE, 2017.

[4] Victor Bisot, Romain Serizel, Slim Essid, and Gaël Richard. Leveraging deep neural networks with nonnegative representations for improved environmental sound classification. In *IEEE International Workshop on Machine Learning for Signal Processing MLSP*, 2017.

[5] Rachel Bittner, Justin Salamon, Mike Tierney, Matthias Mauch, Chris Cannam, and Juan Bello. Medleydb: A multitrack dataset for annotation-intensive mir research.

[6] R.M. Bittner, B. McFee, J. Salamon, P. Li, and J.P. Bello. Deep salience representations for f_0 estimation in polyphonic music. In *18th International Society for Music Information Retrieval Conference*, ISMIR, 2017.

[7] J.J. Bosch, R.M. Bittner, J. Salamon, and E. Gómez. A comparison of melody extraction methods based on source-filter modelling. In *17th International Society for Music Information Retrieval Conference*, ISMIR, 2016.

[8] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014.

[9] Keunwoo Choi, György Fazekas, Kyunghyun Cho, and Mark B. Sandler. A tutorial on deep learning for music information retrieval. *CoRR*, abs/1709.04396, 2017.

[10] François Chollet. keras. `https://github.com/fchollet/keras`, 2015.

[11] J. L. Durrieu, B. David, and G. Richard. A musically motivated mid-level representation for pitch estimation and musical audio source separation. *IEEE Journal of Selected Topics in Signal Processing*, 5(6):1180–1191, Oct 2011.

[12] J. L. Durrieu, G. Richard, B. David, and C. Fevotte. Source/filter model for unsupervised main melody extraction from polyphonic audio signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):564–575, March 2010.

[13] B. Fuentes, A. Liutkus, R. Badeau, and G. Richard. Probabilistic model for main melody extraction using constant-q transform. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5357–5360, March 2012.

[14] A. Graves, A. r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6645–6649, May 2013.

[15] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[16] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 448–456. JMLR.org, 2015.

[17] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[18] Anssi Klapuri. Multiple fundamental frequency estimation by summing harmonic amplitudes. In *ISMIR*, pages 216–221, 2006.

[19] S. Leglaive, R. Hennequin, and R. Badeau. Singing voice detection with deep recurrent neural networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 121–125, April 2015.

[20] B. McFee and J.P. Bello. Structured training for large-vocabulary chord recognition. In *18th International Society for Music Information Retrieval Conference*, ISMIR, 2017.

[21] Brian McFee, Matt McVicar, Stefan Balke, Carl Thomé, Colin Raffel, Dana Lee, Oriol Nieto, Eric Battenberg, Dan Ellis, Ryuichi Yamamoto, Josh Moore, Rachel Bittner, Keunwoo Choi, Pius Friesch, Fabian-Robert Stöter, Vincent Lostanlen, Siddhartha Kumar, Simon Waloschek, Seth, Rimvydas Naktinis, Douglas

Repetto, Curtis "Fjord" Hawthorne, CJ Carr, Waldir Pimenta, Petr Viktorin, Paul Brossier, João Felipe Santos, JackieWu, Erik, and Adrian Holovaty. librosa/librosa: 0.6.1, May 2018.

[22] François Rigaud and Mathieu Radenen. Singing voice melody transcription using deep neural networks. In *ISMIR*, 2016.

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

[24] J. Salamon, E. Gomez, D. P. W. Ellis, and G. Richard. Melody extraction from polyphonic music signals: Approaches, applications, and challenges. *IEEE Signal Processing Magazine*, 31(2):118–134, March 2014.

[25] Emmanuel Vincent, Nancy Bertin, and Roland Badeau. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):528–537, 2010.

[26] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM, 2008.