TISMIR

Voice Assignment in Vocal Quartets Using Deep Learning Models Based on Pitch Salience

RESEARCH

HELENA CUESTA
EMILIA GÓMEZ
*Author affiliations can be found in the back matter of this article

ubiquity press

ABSTRACT

This paper deals with the automatic transcription of four-part, a cappella singing, audio performances. In particular, we exploit an existing, deep-learning based, multiple F0 estimation method and complement it with two neural network architectures for voice assignment (VA) in order to create a music transcription system that converts an input audio mixture into four pitch contours. To train our VA models, we create a novel synthetic dataset by collecting 5381 choral music scores from public-domain music archives, which we make publicly available for further research. We compare the performance of the proposed VA models on different types of input data, as well as to a hidden Markov model-based baseline system. In addition, we assess the generalization capabilities of these models on audio recordings with differing pitch distributions and vocal music styles. Our experiments show that the two proposed models, a CNN and a ConvLSTM, have very similar performance, and both of them outperform the baseline HMM-based system. We also observe a high confusion rate between the alto and tenor voice parts, which commonly have overlapping pitch ranges, while the bass voice has the highest scores in all evaluated scenarios.

CORRESPONDING AUTHOR:

Helena Cuesta

Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain; DAACI Ltd., London, UK

helena@daaci.com

KEYWORDS:

voice assignment; multi-pitch estimation; music information retrieval; vocal quartets; polyphonic vocal music; deep learning

TO CITE THIS ARTICLE:

Cuesta, H., and Gómez, E. (2022). Voice Assignment in Vocal Quartets Using Deep Learning Models Based on Pitch Salience. *Transactions of the International Society for Music Information Retrieval,* 5(1), 99–112. DOI: https://doi.org/10.5334/tismir.121

1 INTRODUCTION

Ensemble singing is an essential part of musical cultures across the world and an essential activity for social entertainment and mental well-being (Clift et al., 2010; Kirsh et al., 2013). It can refer to choral ensembles of any size, from just a few singers, e.g., a quartet, to tens of singers, e.g., a choir. Despite this popularity, *Music Information Retrieval* (MIR) research on polyphonic music has been rather constrained to other types of musical material such as pop/rock (Ryynänen and Klapuri, 2008; Bittner et al., 2017), jazz (Abeßer et al., 2017; Abeßer and Müller, 2021), or piano music (Sigtia et al., 2016; Nakamura et al., 2018) in the past decades. However, several works have recently addressed singers' intonation and interaction in vocal ensembles (Devaney et al., 2012; Dai and Dixon, 2019; Cuesta et al., 2018; Weißet al., 2019), the analysis of vocal unisons (Cuesta et al., 2018, 2019; Chandna et al., 2020), the estimation of multiple fundamental frequency (F0) values from polyphonic vocal performances (Su et al., 2016; Schramm and Benetos, 2017; McLeod et al., 2017; Cuesta et al., 2020), or the separation of singing voices (Gover and Depalle, 2020; Petermann et al., 2020; Sarkar et al., 2020). Most of the tasks mentioned above commonly rely either on analyzing separate audio tracks (stems) for each singer of the ensemble, or on the ground truth individual F0 contours. However, in the context of real vocal ensemble recordings, obtaining such data is not straightforward. On one side, separate audio stems require multitrack datasets, which are very challenging to record in the case of multiple singers performing simultaneously (Cuesta et al., 2018; Rosenzweig et al., 2020). On the other hand, obtaining accurate F0 contours for each voice requires a great manual annotation effort. Alternatively, one could consider source separation algorithms to obtain each voice contribution and then employ monophonic F0 tracking to compute independent F0 trajectories. While a few works address source separation in the context of vocal ensembles (Petermann et al., 2020; Gover and Depalle, 2020; Sarkar et al., 2020) and show promising results, the output isolated voices contain artifacts or leakage from other voices that make the monophonic F0 extraction more challenging. In light of these challenges, advances in automatic transcription and multi-pitch estimation (MPE) systems provide an excellent opportunity to optimize these tasks since, in the ideal case, they strongly reduce the need for manual annotations or separated recordings.

There are several approaches for MPE specifically designed for and evaluated in vocal music: SST-ConceFT (Su et al., 2016), MSINGERS (Schramm and Benetos, 2017), VOCAL4-MP and VOCAL-VA (McLeod et al., 2017), and Late/Deep CNN (Cuesta et al., 2020). However, all but one of these approaches produce a multi-pitch output, which consists of several F0 values per time frame, providing no indication of the singer each pitch belongs to. For these outputs to be used as intermediate representations for the aforementioned tasks, an additional step to assign each predicted pitch to one of the voices in the ensemble is still necessary, a process known as *Voice Assignment* (VA), which converts the multi-pitch output into four (in the case of quartets) F0 trajectories. To the authors' knowledge, VOCAL4-VA is the only approach tackling both tasks—MPE and VA—jointly for four-part vocal ensembles.

An alternative method to obtain independent F0 trajectories for each singer is to address the task of multi-pitch streaming (MPS). MPS is described by Benetos et al. (2019) as grouping estimated pitches or notes into streams, where each stream typically corresponds to one instrument or musical voice. When we combine an MPE system with a VA one, the result is a system that yields independent F0 contours for each source in the input audio mixture just as in MPS.

In this paper, we propose a data-driven pipeline that combines an existing MPE approach with a novel deep learning approach for VA. We select this two-stage pipeline over an entire MPS system because it enables training both steps separately. Consequently, and as we describe in the following sections, we can consider an independent, synthetic dataset to develop the VA module. The proposed system is illustrated in Figure 1: given an input audio mixture of a vocal quartet, we first use Late/Deep CNN to compute a pitch salience representation of the input audio mixture. Late/Deep CNN is a convolutional neural network (CNN) specifically trained with Soprano, Alto, Tenor, Bass (SATB) vocal quartets, which produces a pitch salience representation that can be post-processed and converted to a multi-pitch output. We select this model because it shows the highest performance in terms of MPE in the experiments of Cuesta et al. (2020), as well as being publicly available. Then, we use the output of the CNN as input to the proposed VA approach. In particular, we propose two novel deep learning architectures for VA of four-part vocal music, which take a polyphonic pitch salience representation as input, and produce four separate, monophonic pitch salience representations (cf. Figure 1d), which are subsequently post-processed and converted into four independent F0 trajectories, the final output of the proposed pipeline. We target vocal ensembles with four distinct and simultaneous voice parts, e.g., SATB.

In the presented experiments, we compare the performance of the two proposed architectures, as well as assessing their generalization capabilities to audio material with different pitch ranges and timbre. This work also contributes to state-of-the-art research by providing an open dataset to be exploited for this particular task.

The remainder of this paper is organized as follows. We review the existing literature in Section 2. Then, we provide details about the dataset exploited in our work in

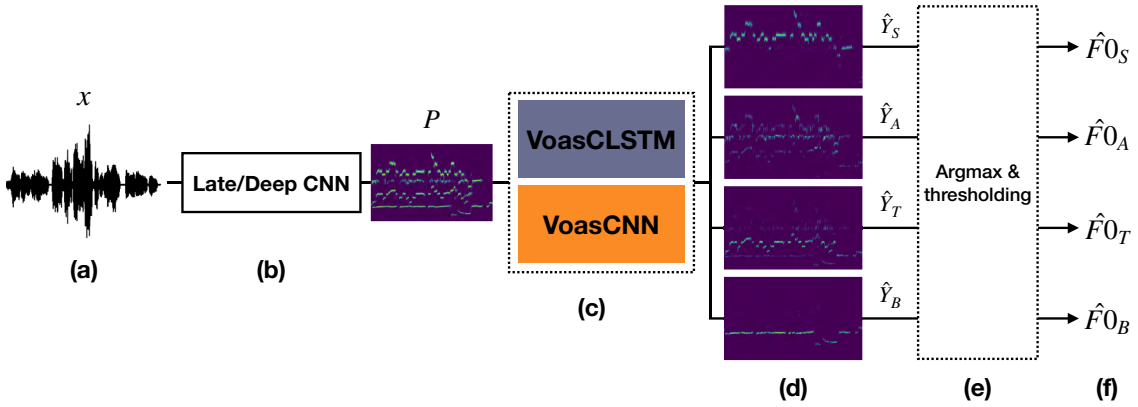


Figure 1: Overview of the proposed system for multi-pitch estimation and voice assignment based on pitch salience representations. **(a)** Input audio SATB mixture. **(b)** Multi-pitch salience estimation using the Late/Deep CNN, which produces the salience representation *P*. **(c)** Voice assignment step with one of the two proposed architectures. **(d)** Four output salience representations, one for each voice in the mixture, output by the VA models. **(e)** Post-processing step consisting of finding the maximum salience bin in $\hat{Y}_v$ and thresholding. **(f)** Output F0 trajectories for each singer.

Section 3, and describe our framework and experiments in Section 4 and 5, respectively. Results are presented in Section 6, while Section 7 contains a discussion and error analysis. We close with the conclusions in Section 8.

2. RELATED WORK

Voice assignment (VA), also referred to as voice separation or voice tracking, is commonly defined as the process of allocating notes of a given piece of music into separate melodic streams (McLeod and Steedman, 2016). However, there is no standard, common definition of the concepts of *voice* or *melody*, so different fields of study, e.g., traditional musicology, music cognition, or computational musicology provide different views on these terms.

The work by Cambouropoulos (2008) provides a broad discussion on what "voice" means and how we can systematically describe the task of VA. In the context of this paper, we refer to a voice as the melodic stream produced by one singer of the ensemble. At this point, it is important to emphasize that most of the work around automatic VA for polyphonic music has analyzed symbolic music representations (MIDI files), while the focus of the proposed VA approach are audio-based pitch salience representations.

Huron (2001) investigates the perceptual principles that derive the rules of voice leading in Western music. He presents six main perceptual principles—toneness, temporal continuity, maximum masking, tonal fusion, pitch proximity, and pitch co-modulation. Although all of them play a crucial role in how humans perceive melodies, we find *pitch proximity* and *temporal continuity* to be the foundations of most literature around VA in music. The pitch proximity principle states that successive notes should maintain a close pitch proximity to be perceived in the same stream. The temporal continuity refers to the idea that a melodic stream should be rather continuous or recurrent, and not have large silent parts in between. A large number of VA approaches follow these two principles (Kilian and Hoos, 2002; Madsen and Widmer, 2006; McLeod and Steedman, 2016; Kirlin and Utgoff, 2005; Gray and Bunescu, 2016), while other research works only consider the pitch proximity principle (Chew and Wu, 2004; Jordanous, 2008).

A group of heuristic-based methods explicitly use musical knowledge to design different local (Kilian and Hoos, 2002; Madsen and Widmer, 2006) and global (Chew and Wu, 2004) cost functions to solve the VA task. A second group of approaches are data-driven, i.e., they exploit data examples to build statistical models that support the segregation of the voices of an input symbolic representation (Jordanous, 2008; Jin and Wang, 2020; Gray & Bunescu, 2016). Other methods combine knowledge- and data-driven techniques (McLeod and Steedman, 2016; Kirlin and Utgoff, 2005). In the latter category, McLeod and Steedman (2016) propose a hidden Markov model (HMM) designed with perceptual principles in mind, the model being trained with MIDI data using grid search for parameter optimization.

All the above-mentioned approaches operate on MIDI-like files, and in most of the cases they are developed and tested on piano music. However, McLeod et al. (2017) propose a probabilistic adaptation of the HMM-based method from McLeod and Steedman (2016) for VA in vocal quartets. Particularly, they follow a procedure similar to our proposed framework: the authors first run an MPE algorithm optimized for polyphonic vocal music, based on spectrogram factorization. Then, the HMM-based model is applied to the MPE outputs to assign each extracted pitch to its corresponding voice. Additionally, they propose to further use the VA output to refine the F0 estimates, which results in a performance boost in both parts. This two-stage approach outputs independent pitch contours, which is equivalent to the

outputs of frame-based MPS, as mentioned in Section 1. MPS has not been widely addressed in the MIR literature, but we find a few existing approaches following different underlying principles. Some studies address the task by employing MPE as a first step and then additionally consider timbral features to separate pitch values into streams via constrained clustering (Duan et al., 2013) or deep spherical clustering (Tanaka et al., 2020). More recently, Lordelo et al. (2021) proposed a data-driven modular pitch-informed classification system that assigns every note to its source via a CNN considering pitch information from an MPE system. Arora and Behera (2015) combine PLCA with hidden Markov random fields (HMRF) to decompose the audio signal and group pitches into separate streams, respectively. Benetos and Dixon (2013) consider the temporal evolution of notes and build spectral templates that model pitch and note states. Then, shift-invariant PLCA is used to stream pitches. Finally, to our knowledge, the work by McLeod et al. (2017) described above is the only existing method for MPS (MPE and VA) in the context of vocal ensembles. The modular systems for MPS we just described are one source of inspiration for the proposed pipeline for MPE and VA.

3 DATA COLLECTION

As mentioned in Section 2, most research on voice assignment has focused on the processing of symbolic music representations, mostly following rule-based approaches, and has mainly addressed piano music. Hence, there are no large-scale open datasets to be exploited for the development of data-driven methods for this task, and in particular for the case of vocal ensembles. The creation of such a dataset, described in this section, is one of the contributions of this work.

3.1 SYNTH-SALIENCE CHORAL SET

Training a data-driven model for VA requires a large-scale, representative dataset, which should be heterogeneous so that the trained model can generalize to different songs and diverse styles of choral music, potentially with varying harmonic relations between voices. Therefore, the training dataset needs to cover a large number of different song styles. Due to the lack of an appropriate dataset for this task, we present here a synthetic dataset built from a large set of choral music scores from public-domain archives, which we convert to our target input and output features: the *Synth-salience Choral Set* (SSCS). The dataset building methodology is detailed as follows.

3.1.1 Public-domain music archive

We collect scores of four-part (SATB) a cappella choral music from the *Choral Public Domain Library* (CPDL)[1] using their API. We assemble a collection of 5381 scores in MusicXML format, which we subsequently convert into MIDI files using the `Music21` Python library (Cuthbert and Ariza, 2010). For training and evaluating our models, we use 75% of the scores for training (4036), 15% for testing (807), and 10% for validation (538).

3.1.2 Pitch salience representation

The proposed VA system relies on a pre-computed time-frequency representation of a music piece. Following the nomenclature of Bittner et al. (2017, 2018), we denote this representation as a pitch salience function. By definition, an "ideal" pitch salience function of a music recording is zero everywhere where there is no perceptible pitch, and has a positive value that reflects the pitches' perceived energy at the frequency bins of the corresponding F0 values. In practice, for a normalized synthetic pitch salience function we assume a value equal to the maximum energy (salience), i.e., 1, in the time-frequency bins that correspond to the notes present in a song, and 0 elsewhere. We can obtain such a synthetic pitch salience representation directly by processing the digital (MusicXML, MIDI) score of a music piece, using the desired time and frequency quantization, i.e., a time-frequency grid. The process is detailed in the following section.

3.1.3 Score to pitch salience

We first convert each MIDI track to an F0 trajectory: a time series with a tuple *(timestamp, F0)* at every time step, with the desired hop size (11 ms) and its corresponding MIDI pitch converted to Hertz. Note that for all time frames that belong to the same single MIDI note, their associated F0 will be the same, i.e., each note is represented by several frames with the same pitch value.

In order to create more realistic synthetic data, with pitch instabilities and noise, we apply some degradation to the F0 trajectories. First, we add some noise frame-wise by drawing random samples from a normal (Gaussian) distribution with standard deviation of five bins (one semitone). This noise adds some variability to the F0 trajectories and reduces the flatness of the MIDI notes, making them look more realistic. However, the transitions between notes are still very abrupt when compared to a real singing voice signal. To overcome this limitation, we apply a median filter with a window size of seven frames (ca.77 ms) that creates more realistic note transitions, i.e., smoother, while keeping the roughness within the notes. This process is not optimal because pitch variations and note transitions in real singing voice follow some patterns, e.g., vibrato or slides. However, we conduct some experiments (see Section 5.3) and find this simple method to be effective enough to account for such variations in real recordings. Then, we map each pair *(timestamp, F0)* to their corresponding

time-frequency bin in the grid and assign them a magnitude of one, while setting to zero all other bins. Finally, to account for possible imprecision in the predictions and following Bittner et al. (2017), we apply Gaussian blur with a standard deviation of 1 bin in the direction of the frequency axis. We store each of these pitch salience representations as CSV files. Figure 2 shows an example from our synthetic dataset, displaying the input salience function (bottom pane) and the four voice-specific outputs (top panes).

Following reproducible research practices, SSCS is publicly available.[2] See the supplementary file for more information about the structure of SSCS. Note that this dataset contains the input/output features we use in our study, i.e., salience functions, and not audio files nor scores.

3.2 AUDIO RECORDINGS

In our experiments, we use an additional set of audio files to evaluate the proposed framework on real vocal quartet recordings. In particular, we use the Barbershop Quartets Dataset (BSQ), which is a multi-track collection of 26 songs (ca.42 min of audio) performed by an all-male barbershop quartet and recorded with individual microphones. Additionally, the dataset contains automatically extracted pitch trajectories for each voice of the quartet. This dataset was used by McLeod et al. (2017) in their experiments for MPE and VA. Furthermore, we employ the Cantoría dataset, which comprises multi-track recordings of a professional SATB vocal quartet performing 11 songs (ca. 36 min of audio), and was introduced by Cuesta (2022). This collection also contains automatically extracted pitch trajectories for each singer and it is publicly available.[3]

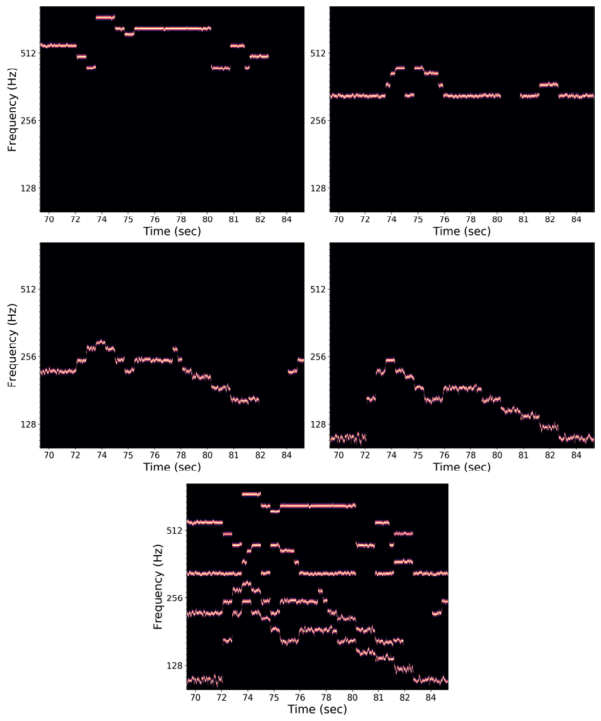


Figure 2: Example input/output data from SSCS dataset. The first four panes show an excerpt of each synthetic Y_v, and the bottom pane displays the input mixture, P.

4. METHODOLOGY

In this work, we propose two deep learning architectures to solve the VA task: VoasCNN and VoasCLSTM, illustrated in Figure 3. Both architectures build upon the multiple F0 estimation CNN models described by Cuesta et al. (2020), i.e., they take as input the output of the CNN. In this section we first describe the input and output features, and then present the two network architectures. Afterwards, we detail the post-processing steps applied to the output of the networks.

4.1 INPUT AND OUTPUT FEATURES

The input to our VA architectures is a pitch salience representation of a polyphonic audio recording, $P \in [0,1]^{F \times T}$, a 2-D array where F is the number of frequency bins in the time-frequency grid, and T corresponds to the number of time frames. We use a fixed time-frequency grid with a hop size of 11 ms and 360 frequency bins. The frequency axis covers 6 octaves with a minimum frequency f_{min} = 32.7 Hz, and a resolution of 20 cents per bin, matching the feature dimensions of the MPE model we use as a first step. We denote the output representations, i.e., the salience functions for each voice part, as $Y_v \in [0,1]^{F \times T}$, where $v \in \{S, A, T, B\}$, and each Y_v has the same size as P.

To generate the input and output data, we consider the choral scores from SCSS (Section 3.1). These scores, in MIDI format, contain multiple tracks, each corresponding to one specific voice. Hence, we first process each track of the score separately to compute the four output representations (targets), Y_v, each of which contains only one voice part. Then, we calculate one input representation that contains all four voices:

$$P = Y_S + Y_A + Y_T + Y_B \tag{1}$$

Note that when two voices sing the same note simultaneously, the corresponding time-frequency bins in P may be larger than 1. To maintain the range [0,1], we set these values to 1. This process discards information from unisons in the input representation, but it is preserved in the output targets, where both voices will have high salience in the corresponding bins. With the synthetic data we calculate P explicitly, mapping the score to the time-frequency grid; however, in the actual system pipeline with an audio mixture as input, P corresponds to the pitch salience function obtained at the output of the last layer of Late/Deep CNN (Cuesta et al., 2020).

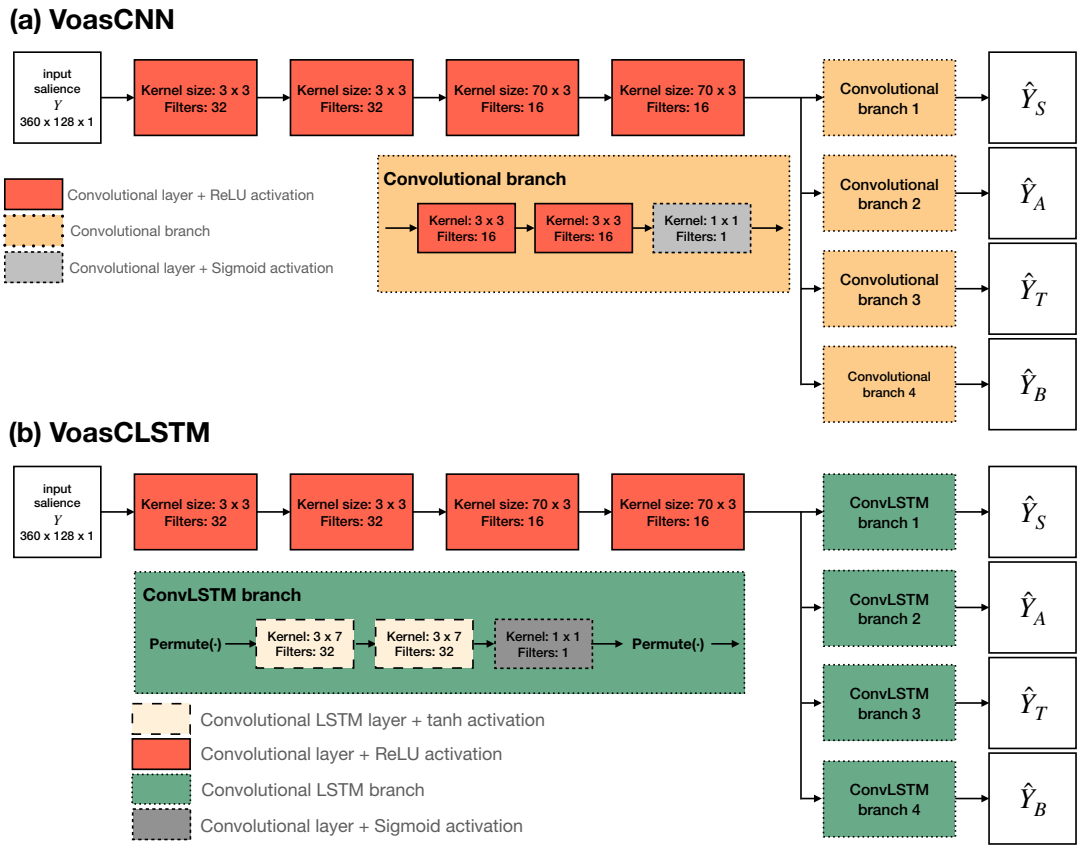


Figure 3: Proposed network architectures. **(a) VoasCNN** is a fully convolutional network with a shared first stage, and four separate branches in the second stage (convolutional branches). **(b) VoasCLSTM** is a network with a first stage of convolutional layers, followed by four separate branches in the second stage with convolutional LSTM layers (ConvLSTM branches). All convolutional layers are preceded by batch normalization.

4.2 VoasCNN

VoasCNN is designed as a fully convolutional architecture with the goal to consider the pitch proximity principle, so that time-frequency bins close in pitch are assigned to the same voice. At the same time, we expect the network to learn specific patterns for unisons and voice crossings, which especially happen between contiguous voices with overlapping pitch ranges. The input to VoasCNN (and, consequently, the output) are patches of the pitch salience function of size (360×128), which cover the full frequency axis (see Section 4.1), and ca. 1.5 seconds of the input audio signal, sampled at 22050 Hz.

VoasCNN has two stages: the first one is composed of four convolutional layers with 32 (3×3)[4], 32 (3×3), 16 (70×3), and 16 (70×3) filters, respectively. Note that the last two layers of this first stage employ vertical filters in the frequency dimension which cover slightly more than one octave, aiming to capture harmonic relationships between the voices in this range. Batch normalization precedes all layers, and all of them use rectified linear units (ReLU) as activation. In the second stage, the network creates four separate branches that operate independently, i.e., one for each voice. Each of these branches has two convolutional layers with 16 (3×3) filters, and a final layer with a sigmoid function as activation to map the output of each time-frequency bin to the range [0,1], obtaining $\hat{Y}_v$.

4.3 VoasCLSTM

The second proposed architecture is VoasCLSTM, a convolutional long short-term memory (ConvLSTM) network. Long short-term memory (LSTM) networks are a type of recurrent neural network (RNN) that use a set of gate units to control which information from a past state should be kept for the current state (Hochreiter and Schmidhuber, 1997). In the context of an LSTM, the input stream encoding information from the past, $\boldsymbol{i}^{(t)}$, can be formulated in a compact form as follows for one specific layer:

$$\boldsymbol{i}^{(t)} = \sigma\left(\boldsymbol{W}\boldsymbol{x}^{(t)} + \boldsymbol{W}_{rec}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}\right) \quad (2)$$

where $\boldsymbol{x}^{(t)}$ denotes the input at time t, $\boldsymbol{W}_{rec}$ refers to the recurrent weights encoding temporal dependencies, and $\boldsymbol{h}^{(t-1)}$ denotes the output of the layer at time instant t-1. We can think of $\boldsymbol{W}_{rec}$ as the weights that decide the amount of information from the past that is considered for the current prediction.

The ConvLSTM architecture is a special type of LSTM first introduced by Shi et al. (2015) for the task of precipitation nowcasting, a spatiotemporal sequence problem that consists of forecasting future radar maps using previously

observed radar echo sequences. Intuitively, we can think of ConvLSTM layers as a combination of the properties of convolutional layers, i.e., modeling "spatial" information (frequency-related information in this context), and those of LSTMs, i.e., modeling "temporal" information. Consequently, ConvLSTMs are suitable for tasks where data have both dimensions: spatial and temporal. For example, in the original example of precipitation nowcasting, input data are radar echo maps (2D images, spatial data), and they are captured at every time step, creating the temporal dimension. In a ConvLSTM, the input-to-state and the state-to-state transitions have convolutional structures to handle spatial data in a more efficient way. Following Equation (2), in the context of a ConvLSTM, $\boldsymbol{i}^{(t)}$ can be formulated as:

$$\boldsymbol{i}^{(t)} = \sigma\left(\boldsymbol{W} * \boldsymbol{x}^{(t)} + \boldsymbol{W}_{rec} * \boldsymbol{h}^{(t-1)} + \boldsymbol{b}\right) \tag{3}$$

where $*$ denotes a convolutional operation, which replaces the point-wise multiplications from Equation (2).

In the context of the VA task, we believe that adding recurrence should effectively support the separation of melodic streams into their underlying voices, using the information from past frames as an indicator for the time continuity principle. The input to VoasCLSTM (and, consequently, the output) are patches of the pitch salience function of size (360, 128), which cover the full frequency axis (see Section 4.1) and ca. 1.5 seconds of the input audio signal. The proposed VoasCLSTM consists of an initial branch with four convolutional layers, just as in VoasCNN, with 32 (3×3), 32 (3×3), 16 (70×3), and 16 (70×3) filters, respectively. All convolutional layers use ReLU activation and are preceded by batch normalization. Then, the network is divided into four separate branches. Each of these branches is made of two ConvLSTM layers with 32 (3×7) filters each, tanh activation function, and hard sigmoid as the recurrent activation. We choose these activations based on the analysis of Elsayed et al. (2019), who compared multiple activation functions for a video prediction task and found the combination of hard sigmoid as a recurrent activation and tanh as standard activation to obtain the best performances. The last layer of each branch is a convolutional layer that uses a sigmoid activation function, obtaining $\hat{Y}_v$.

4.4 TRAINING

Both networks are trained to minimize the cross-entropy loss, $L(Y,\hat{Y})$, calculated as the sum of the cross-entropy between the target representations, Y_v, and the predictions, $\hat{Y}_v$, for each voice:

$$L\left(Y,\hat{Y}\right) = \sum_v -Y_v \log\left(\hat{Y}_v\right) - (1-Y_v)\log\left(1-\hat{Y}_v\right) \tag{4}$$

We use the Adam optimizer (Kingma and Ba, 2014) with an initial learning rate of 0.005, and we train for 100 epochs, using the validation set for early-stopping with a patience of 20 epochs.

4.5 POST-PROCESSING

The last step of our VA pipeline consists of a two-stage process including locating maximum salience bins and thresholding. In particular, for each time frame n, we first locate the frequency bin of $\hat{Y}_v[n]$ with the highest salience. Second, the selected bin is converted into its corresponding F0 value if the salience is above a threshold. The thresholding step filters out spurious, low-salience bins, which is particularly helpful for the unvoiced frames where the salience representations may show some very low salience. The threshold is optimized on the validation set after training. We calculate one optimal threshold for each of the voices as the average of the ones that maximize the *Overall Accuracy* (OA) of each individual F0 trajectory and voice for all validation examples. The OA measures the percentage of predictions made by the algorithm that are correct both in terms of F0 and voicing.

5. EXPERIMENTAL SETUP

5.1 EVALUATION METRICS

We evaluate the proposed models for VA using the *F-Score* (F), a frame-based standard evaluation metric widely used for (multi)-pitch estimation. We use the implementation from the multi-pitch class in the `mir_eval` library (Raffel et al., 2014). We consider this metric both for per-voice evaluation, i.e., assessing the output of each voice individually as monophonic streams, and for multi-pitch evaluation, i.e., assessing the combination of the outputs of each voice as a multi-pitch stream. The main difference is that for per-voice evaluations, the reference contains at most one F0 value per frame, while multiple F0 values can be present in the multi-pitch reference.

For all metrics, a predicted pitch is considered correct if it is within a half semitone of the correct pitch in the reference. For the per-voice evaluations we compare our separated F0 trajectories to the F0 trajectories of each individual voice as ground truth (GT). For the multi-pitch evaluations, we combine the F0 trajectories of the four voices into a single multi-pitch stream, for both the predictions and the GT. In the case of the synthetic dataset, the GT F0 trajectories come directly from the score pitches. For real recordings, the dataset includes F0 trajectories for each singer in the ensemble. In terms of notation, in per-voice evaluations we use F_v, where the subindex v indicates the voice part; in multi-pitch evaluations we use F_{MPE}.

We want to point out that a common evaluation metric in the related research mentioned in Section 2 is the *Average Voice Consistency* (AVC). We decided to exclude this metric from the evaluation because it is based on notes, while our results are frame-based.

5.2 EXPERIMENT 1: ARCHITECTURE COMPARISON

In this first experiment we analyze the suitability of the designed architectures to tackle the VA task. We consider the full synthetic dataset, split into training-validation-test subsets, to train and evaluate VoasCNN and VoasCLSTM, thus assessing the effect of using a fully convolutional network (VoasCNN) as compared to adding recurrence (VoasCLSTM). We use the validation set to optimize the threshold for the post-processing step: we calculate four optimal thresholds, i.e., one per voice, each of which maximizes the average OA on the validation set for their corresponding voice. We obtain very similar optimal thresholds for both models; specifically, for VoasCNN: 0.23, 0.17, 0.15, 0.17, while VoasCLSTM obtains: 0.29, 0.20, 0.17, 0.23, for soprano, alto, tenor, and bass voices, respectively.

As a baseline for this experiment we use the VA HMM-based system by McLeod and Steedman (2016), who provided us with an adaptation of their model that runs on similar input data as our models, i.e., pitch salience-like representations. While our proposed VA models take between 20 and 30 seconds to run for one example on a CPU machine (roughly between one and five seconds on a GPU), our baseline is computationally more demanding, and can take over 20 minutes to calculate the output of one full example on the same machine.

5.3 EXPERIMENT 2: DATA DEGRADATION

The goal of this experiment is to assess the effect of the data degradation process, specifically for our use case where the input signals are polyphonic singing audio recordings, and not MIDI files, which contain more noise and pitch instabilities. Our main hypothesis is that we will observe a drop in the performance when the model operates on an audio recording, as compared to a synthetic input. In particular, this experiment considers SATB mixtures from the Cantoría dataset, presented in Section 3.2. We evaluate the music transcription system outlined in Figure 1 consisting of an MPE algorithm (pre-trained Late/Deep CNN, publicly available) followed by a VA module. Based on experiment 1, we only run this second experiment on one of the two proposed architectures; in particular, since they yield very similar performances as shown in Section 6.1, we select the VoasCNN because it is faster at inference time.

We train VoasCNN with two different variants of the synthetic dataset: we first consider the synthetic dataset directly created from the choral scores, i.e. "clean" dataset, **C-VoasCNN**; second, we consider a "degraded" dataset with noise and median filtering (see Section 4.1), **D-VoasCNN**. We then combine these two model variants with the MPE algorithm and evaluate the full transcription systems with real audio recordings (Cantoría dataset).

We report the results of C-VoasCNN and D-VoasCNN averaged across all the songs per-voice, in terms of multi-pitch estimation results before and after the VA process. This comparison between pre- and post-VA results in terms of multi-pitch estimation provides insights as to the amount of error introduced by the VA stage: in the ideal case, where no information is lost, the MPE results pre- and post-VA should be equivalent. However, if the post-VA results are worse than the pre-VA ones, we can consider that, even if the VA step adds value for further tasks as it provides separated voices, it might come at the cost of a lower overall performance.

5.4 EXPERIMENT 3: MODEL GENERALIZATION

In this last experiment we evaluate the complete automatic transcription framework by combining MPE and VA tasks to assess the generalization capabilities of our models, trained with synthetic data, to real recordings in a different pitch range than the training set. This is done by running the whole pipeline on the Barbershop Quartets dataset (BSQ) (cf. Section 3.2). By using this material, we can evaluate how our models generalize from synthetic to audio recordings, as well as to a vocal ensemble where the singers' tessitura differs from the training material, i.e., we trained with SATB data, while the BSQ dataset contains only-male voices, thus lower pitches in general.

In addition, evaluating with the BSQ recordings is beneficial for two further reasons: first, because they are used by Schramm and Benetos (2017) and McLeod et al. (2017), thus allowing for direct comparison with their systems: their MPE systems (MSINGERS and VOCAL4-MP, respectively) and the full MPE plus VA method (VOCAL4-VA) from the latter. The second reason is because Cuesta et al. (2020) provide a version of Late/Deep CNN trained with all datasets except the BSQ. Therefore, the input recording is also unseen for the MPE model, making the entire pipeline run on an independent input. In this experiment, we consider this version of Late/Deep CNN instead of the full version utilized in Experiment 2. We additionally compare the results to the combination of Late/Deep CNN and the HMM-based VA baseline. We aim to explore the behaviour of the VA model when the input salience function is noisy and the pitch is more unstable—as opposed to the stable pitch that we observe in the synthetic dataset examples.

6 RESULTS

6.1 ARCHITECTURE COMPARISON RESULTS

Figure 4 depicts the average results for experiment 1. The first aspect we observe is that VoasCNN and VoasCLSTM show an almost equivalent performance for all voices, while outperforming the HMM baseline by a large margin (+10% in average F-Score). While VoasCLSTM shows a slightly better performance than its fully convolutional equivalent (+1.5% in average F-Score and Recall, +1% in

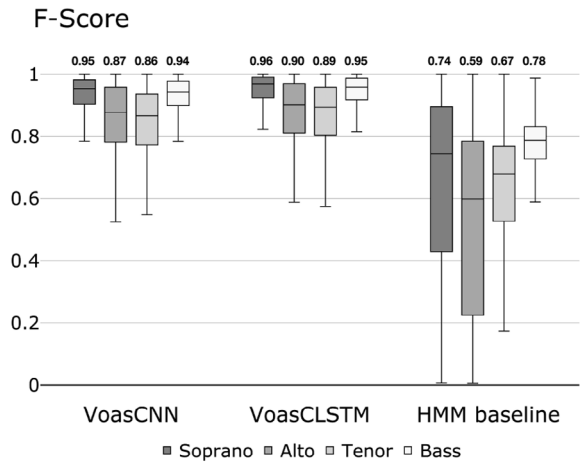


Figure 4: Experiment 1 results. Boxplots with the evaluation results (per-voice F-Score) of the two proposed models (VoasCNN and VoasCLSTM) and the HMM-based baseline on the synthetic test set. The horizontal line inside the boxes shows the median of the distribution, and the numbers above each box indicate the corresponding numerical value.

average Precision), the difference is not large enough to conclude that it is better than VoasCNN for the task.

In addition, we observe a lower performance in alto and tenor voices with all models, which we assume is due to these voices having overlapping pitch ranges. Soprano and bass parts are at the high and low ends, thus being easier for the model to decide where to classify them when in case of dubious passages, i.e., the lowest F0 always goes to the bass, and the highest to the soprano, although this is not necessarily always true.

6.2 DATA DEGRADATION RESULTS

In experiment 2, we assessed the combined MPE plus VA pipeline on nine songs from the Cantoría dataset, using the two VoasCNN variants: without (C-VoasCNN) and with (D-VoasCNN) data degradation. Table 1 contains the per-voice results in terms of F-Score, which we compute for each voice and model, as well as the combined multi-pitch F-Score (F_{MPE}) before (MPE results directly from Late/Deep CNN) and after the assignment. We first observe that these results follow a similar trend to those in experiment 1 from the perspective of the different voice parts: in relative numbers, both model variants perform better in soprano and bass cases, while they have more difficulties with alto and tenor parts.

Regarding the main focus of the experiment, these results suggest that C-VoasCNN is more suitable than D-VoasCNN for soprano and bass voices, while we observe the opposite behaviour for alto and tenor voices. Interestingly, the soprano and bass results are closer to the multi-pitch results, while they largely differ for the alto and tenor voices. Since the multi-pitch evaluation only checks whether a pitch is present or not, this finding suggests that alto and tenor frequencies are misclassified, i.e., the pitches are most likely to be assigned to the wrong voice.

Voice/Model	C-VoasCNN	D-VoasCNN	L/D CNN
$F_{Soprano}$	**0.77 *(0.05)***	0.73 *(0.06)*	–
F_{Alto}	0.51 *(0.07)*	**0.56 *(0.10)***	–
F_{Tenor}	0.54 *(0.05)*	**0.56 *(0.08)***	–
F_{Bass}	**0.76 *(0.06)***	0.71 *(0.06)*	–
F_{MPE}	0.75 *(0.03)*	0.77 *(0.03)*	**0.85 *(0.03)***

Table 1: Data degradation experiment results: voice-specific F-Score obtained with C-VoasCNN and D-VoasCNN (F_{voice}), and multi-pitch F-Score (F_{MPE}) calculated by combining the four assigned trajectories (post-VA). We additionally report the average F_{MPE} obtained with L/D CNN (Late/Deep CNN, pre-VA). The best result for each voice is highlighted in bold and standard deviations are displayed in italics.

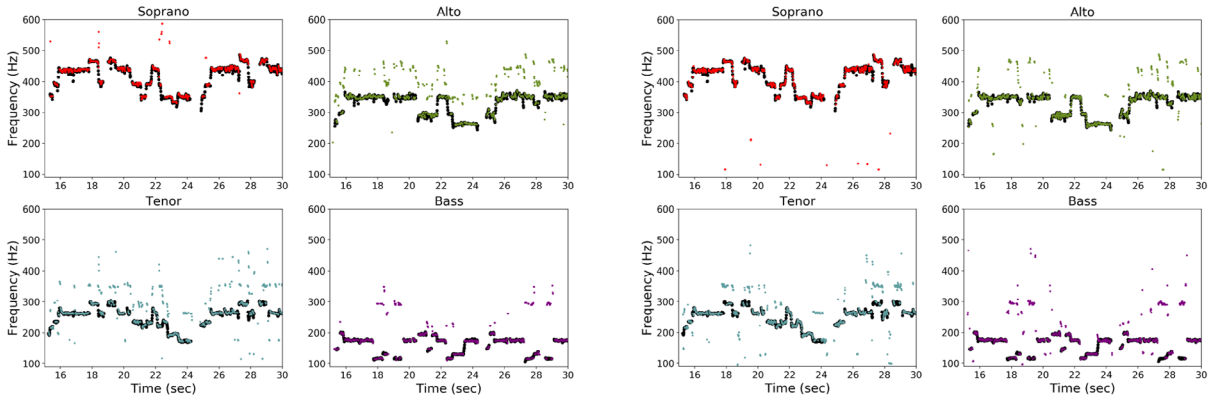
When we focus on the comparison between pre- and post-VA scenarios in terms of multi-pitch metrics (last row), we find a difference of 8–10% in average F-Score for both VA models with respect to the MPE alone. In practice, this means that almost all (roughly 90%) information that the polyphonic salience function contains at the output of the Late/Deep CNN is preserved when we combine the four outputs of VoasCNN. However, the numbers are significantly lower in the per-voice evaluation, which we associate to the voice confusion errors mentioned above.

Figure 5a depicts an example of the outputs of Late/Deep CNN + C-VoasCNN, while the output in Figure 5b uses D-VoasCNN, both for the same excerpt of the song *Virgen Bendita sin par*. This example illustrates some of the voice confusions, especially present in alto and tenor voices, and helps detecting potential recurrent errors. If we focus on the alto part, we observe how it has several spurious peaks that belong to the soprano voice; looking at the tenor voice, we also observe a significant increase of misplaced peaks that belong to the alto voice. An additional observation is that we find more spurious peaks in the bass voice with D-VoasCNN than with C-VoasCNN; similarly, some lower pitch values are assigned to the soprano with the D-VoasCNN, while most mistakes with the C-VoasCNN come from higher pitch values.

This experiment provides some insights about the use of data degradation in our synthetic dataset, although the results are not conclusive enough. While degradation does not consistently improve the performance of VoasCNN, it does seem to help with the alto and tenor parts. Since these are the two most challenging voices in our task, we use the degraded version of the SSCS to train VoasCLSTM for our last experiment.

6.3 MODEL GENERALIZATION RESULTS

Table 2 summarizes the results of the generalization experiment: first, we report the MPE results pre-VA for reference with MSINGERS, VOCAL4-MP and Late/Deep CNN (L/D CNN). Then, the post-VA results with several configurations: Late/Deep CNN combined with our



(a) Late/Deep CNN + **C-VoasCNN** **(b)** Late/Deep CNN + **D-VoasCNN**

Figure 5: Post-VA F0 outputs (color) vs. F0 ground truth (black) for an excerpt of *Virgen Bendita sin par* from the Cantoría dataset. The thresholds we use for this experiment are optimized per-voice on the validation set.

Model generalization Barbershop Quartets Dataset					
Model	$\mathbf{F}_{Soprano}$	$\mathbf{F}_{Alto}$	$\mathbf{F}_{Tenor}$	$\mathbf{F}_{Bass}$	$\mathbf{F}_{MPE}$
MSINGERS	–	–	–	–	0.71 *(0.06)*
VOCAL4-MP	–	–	–	–	0.59 *(0.05)*
L/D CNN	–	–	–	–	**0.84 *(0.03)***
VOCAL4-VA	0.42 *(0.18)*	0.34 *(0.16)*	0.35 *(0.16)*	0.84 *(0.06)*	0.75 *(0.06)*
L/D + HMM	0.68 *(0.12)*	0.43 *(0.16)*	0.40 *(0.18)*	0.66 *(0.15)*	0.77 *(0.06)*
L/D + C-VoasCNN	0.75 *(0.12)*	0.50 *(0.16)*	0.57 *(0.13)*	**0.89 *(0.04)***	**0.84 *(0.04)***
L/D + D-VoasCNN	**0.76 *(0.09)***	**0.59 *(0.15)***	0.57 *(0.14)*	0.85 *(0.05)*	**0.84 *(0.04)***
L/D + D-VoasCLSTM	0.65 *(0.15)*	0.54 *(0.17)*	**0.59 *(0.14)***	0.84 *(0.05)*	0.83 *(0.05)*

Table 2: Evaluation results of the generalization experiment on the BSQ. *L/D* stands for Late/Deep, and HMM refers to the HMM-based baseline for VA. The first three rows correspond to MPE models, thus we only report MPE results. Rows 4 to 8 show MPE+VA results with two baselines (VOCAL4-VA and L/D + HMM) and our proposed models (L/D + C-/D-VoasCNN and D-VoasCLSTM). Standard deviations are indicated in italics, and the best performance for each voice and configuration are highlighted in boldface.

C-VoasCNN, D-VoasCNN and D-VoasCLSTM, VOCAL4-VA system, and the combination of Late/Deep CNN with our baseline HMM-based system on the BSQ recordings. The results with MSINGERS, VOCAL4-MP, Late/Deep CNN, and VOCAL4-VA are taken directly from their original papers, since the input data is the same.

These results provide several insights into the generalization capabilities of our model. First, we observe that the results with C-VoasCNN and D-VoasCNN are very similar for three (STB) of the four voices, while D-VoasCNN scores a 9% higher average F-Score in the alto voice. Looking at VoasCLSTM, it shows a slightly better performance for the tenor voice (+2%), while it achieves inferior results for the other voices, when compared to both VoasCNN variants. Comparing our VA models to the baselines, we find they outperform them in all voice parts, the performance increase being larger when compared to VOCAL4-VA than to Late/Deep combined with the HMM baseline. However, we find the same pattern with all systems: better results for soprano and bass than alto and tenor.

We report the MPE results with and without VA in the last column. In this case we observe that C-VoasCNN and D-VoasCNN produce the best results post-VA, achieving the same average F-Score as Late/Deep CNN alone. Late/Deep CNN with D-VoasCLSTM follows with an equivalent performance (–1% in average F-Score), while the combination of Late/Deep with the HMM baseline performs worse (–7% with respect to Late/Deep CNN).

These experiments show that our models are capable of generalizing to audio recordings even if they are trained exclusively on synthetic data from MIDI. When compared to the results from experiment 1, we observe a performance drop in the BSQ case with respect to the evaluation on synthetic data, which is expected given the synthetic training data. Nevertheless, our models significantly outperform the baselines on the same real audio recordings. In addition, these results also show that our model is capable of generalizing to input audio mixtures where singers have a different pitch range (i.e., BSQ), compared to the Western standard choir configuration, i.e., SATB (cf. Table 1).

7. DISCUSSION AND ERROR ANALYSIS

This section starts by comparing the outputs of the models on the same song we considered in Figure 5. More specifically, we compare the output of D-VoasCNN (Figure 5b) and the output of D-VoasCLSTM, depicted in Figure 6. Both figures look similar overall (a large part of the melodies are correctly predicted in both cases), which agrees with the results from our experiments. However, we observe that the outputs of VoasCLSTM contain less jumps to other voices, showing a tendency to preserve time continuity of the pitch contours better. For instance, the spurious peaks in lower frequencies from D-VoasCNN's soprano output disappear with VoasCLSTM and similarly for bass, where the erroneous F0 values in higher frequencies are greatly reduced. For alto and tenor the effect is the same, although the number of remaining spurious peaks is larger. We find one interesting difference between the two models' behaviour in the last three seconds of the tenor voice: while VoasCNN mistakenly predicts alto notes in the tenor voice, VoasCLSTM does a much better job in this same excerpt. Although this is only one short example, it shows the potential benefits of adding recurrence in the network for preserving time continuity. However, given that the differences are not very large, more research on the optimal design parameters for the ConvLSTM is necessary for more effectiveness. Besides the network type, the error analysis reveals that imposing some continuity constraints on the outputs via further post-processing should improve the results. Alternatively, replacing the current post-processing with Viterbi decoding to find the most likely sequence for each voice would also lead to improved results with better time continuity. Hence, future research on the topic could explore these methods further.

In the second part of this discussion, we briefly demonstrate the models' performances on the song *Riu riu chiu* from Cantoría, which has alternating bass *solo* and quartet passages. An inspection of the results revealed that this scenario is especially challenging

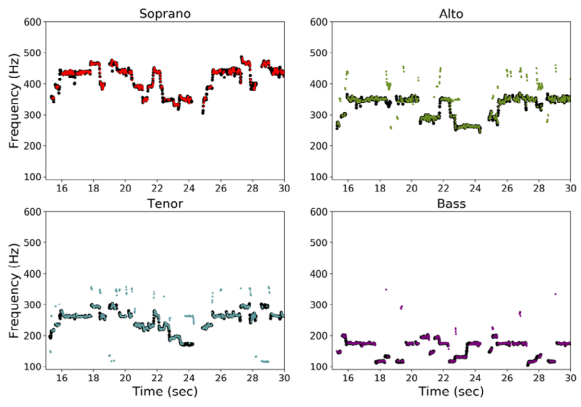


Figure 6: Excerpt of the output of Late/Deep CNN + D-VoasCLSTM on *Virgen Bendita sin par* from Cantoría dataset.

for VA models: it is very likely that every time a solo takes place, the other voices' trajectories are lost. We evaluate the performance of both models considering monophonic F0 estimation evaluation metrics and find two main tendencies: VoasCNN is better at assigning the correct F0s to the correct voices (higher pitch accuracy in general), but VoasCLSTM shows smaller Voicing False Alarm (VFA),[5] indicating a tendency to reduce the number of F0s assigned to the wrong voice. In this context, these results show that VoasCNN is more likely to predict the solo melody in the wrong voice than VoasCLSTM. For instance, the bass solo melody is predicted by VoasCNN in the bass, tenor, and alto voices, yielding VFA of 52% and 49% for the last two, respectively. VoasCLSTM predicts the silent passages correctly for the alto, decreasing the VFA to 10%. Overall, this example shows the tendency of VoasCLSTM to model pitch trajectories' continuity better than VoasCNN, since it can keep track of voices better in the presence of solos. However, the presented numerical results and output examples show that the models' performance can largely improve, especially for the inner voices. This last example additionally shows that the proposed networks are not constrained to assigning each bin to only one voice, which enables the correct prediction of unisons but makes models more susceptible to producing this type of error in the presence of solos. One idea to address this situation would be adding voice-wise weights to the training loss to penalize multiple assignments of the same bin. This would complicate the correct prediction of unisons, which would be penalized but still possible, and it has the potential to significantly improve the performance on situations as the one we presented here.

8. CONCLUSION

In this paper, we have presented and evaluated two novel deep learning based models for voice assignment (VA).[6] Combined with an existing deep learning model for multiple F0 estimation (MPE), they constitute a full framework for audio to pitch contours for four-part a cappella singing recordings. To our knowledge, our work is the first attempt to use deep neural networks to approach the VA task. The two proposed network architectures operate on the output of the MPE system—a pitch salience representation of the input audio. Then, they output four independent pitch salience representations, each of which contains only one melodic source. We first proposed VoasCNN, a fully convolutional architecture (see Figure 3a) that aims at considering the pitch proximity principle. Second, we proposed VoasCLSTM, a convolutional LSTM (ConvLSTM) architecture (see Figure 3b) that combines the properties

of CNNs with the properties of LSTMs in a network that aims at considering the pitch proximity as well as the time continuity principles.

We conducted several experiments to evaluate our models on a novel synthetic dataset for the task (Synth-salience Choral Set, SSCS) and on two different sets of four-part a cappella audio recordings (BSQ and Cantoría). Our experiments show an equivalent performance of both architectures on synthetic data, while VoasCNN slightly outperforms VoasCLSTM when the input is a real audio recording. We additionally assessed the effect of the proposed degradation of the synthetic dataset. We found VoasCNN trained on "clean" data to perform slightly better than on "degraded" data for two of the four SATB voices, but larger scale experiments would be helpful to draw final conclusions. Moreover, the proposed pipeline outperformed the HMM-based baselines both on the synthetic test set and on real audio recordings. Besides, we observed similar trends for all models and baselines: alto and tenor voices obtain poorer results than soprano and bass.

While this work focused on vocal recordings, it has the potential to be trained with other types of music recordings and be considered for MPS in other contexts where the input recordings contain multiple simultaneous melodic lines. Furthermore, besides the post-processing aspects discussed in Section 7, one direction to expand this work further is the study of unisons: how the proposed models behave when two voices coincide at the same note at the same time. While this is very common in choral music, we did not look into this specifically in this work, although we anticipate it is very challenging for the models to detect such cases and account for them. Moreover, future work could also create different synthetic training data by generating audio recordings from scores employing singing synthesis techniques, and calculating the pitch salience representations via MPE algorithms.

NOTES

1 http://cpdl.org/.

2 The Synth-salience Choral Set is hosted on Zenodo.

3 The Cantoría Dataset is hosted on Zenodo.

4 In the notation $F(n \times m)$, F indicates the number of filters in one layer and $n \times m$ refers to the size of the convolutional kernels in the layer.

5 Voicing False Alarm (VFA) measures the proportion of predicted F0s in voiced frames that are unvoiced in the reference.

6 Pre-trained models, code, and data splits are available from this Github repository.

SUPPLEMENTARY FILE

The supplementary file for this article is a PDF with further information about the Synth-salience Choral Set structure. DOI: https://doi.org/10.5334/tismir.121.s1

ACKNOWLEDGEMENTS

This work is partially supported by the European Commission under the TROMPA project (H2020 770376), the Spanish Ministry of Science and Innovation under the Musical AI project (PID2019-111403GB-I00), and by AGAUR (Generalitat de Catalunya) through an FI Predoctoral Grant (2018FI-B01015). The authors would like to thank Dr. Andrew McLeod for adapting and sharing their VA method to be used as baseline for the presented experiments, and the four reviewers for their valuable feedback to improve the quality of this manuscript.

COMPETING INTERESTS

Emilia Gómez is a co-Editor in Chief of the Transactions of the International Society for Music Information Retrieval. She had no involvement in the review and editorial processing of this article. The authors have no other competing interests to declare.

AUTHOR CONTRIBUTIONS

Both authors contributed to conception and design of the experiments. HC trained the voice assignment models and conducted the experiments. HC and EG wrote the manuscript and its revised version, and approved the submitted version.

AUTHOR AFFILIATIONS

Helena Cuesta orcid.org/0000-0001-8531-4487
Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain; DAACI Ltd., London, UK

Emilia Gómez orcid.org/0000-0003-4983-3989
Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain; European Commission, Joint Research Centre, Seville, Spain

REFERENCES

Abeßer, J., Balke, S., Frieler, K., Pfleiderer, M., and **Müller, M.** (2017). Deep learning for jazz walking bass transcription. In *Proceedings of the AES International Conference on Semantic Audio*, pages 202–209, Erlangen, Germany.

Abeßer, J. and **Müller, M.** (2021). Jazz bass transcription using a U-Net architecture. *Electronics*, 10(6). DOI: https://doi.org/10.3390/electronics10060670

Arora, V. and **Behera, L.** (2015). Multiple F0 estimation and source clustering of polyphonic music audio using PLCA and HMRFs. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(2):278–287. DOI: https://doi.org/10.1109/TASLP.2014.2387388

Benetos, E. and **Dixon, S.** (2013). Multiple-instrument polyphonic music transcription using a temporally

constrained shift-invariant model. *The Journal of the Acoustical Society of America*, 133(3):1727–1741. DOI: https://doi.org/10.1121/1.4790351

Benetos, E., Dixon, S., Duan, Z., and **Ewert, S.** (2019). Automatic music transcription: An overview. *IEEE Signal Processing Magazine*, 36(1):20–30. DOI: https://doi.org/10.1109/MSP.2018.2869928

Bittner, R. M., McFee, B., and **Bello, J. P.** (2018). Multitask learning for fundamental frequency estimation in music. *ArXiv*, abs/1809.00381.

Bittner, R. M., McFee, B., Salamon, J., Li, P., and **Bello, J. P.** (2017). Deep salience representations for F0 tracking in polyphonic music. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 63–70, Suzhou, China.

Cambouropoulos, E. (2008). Voice and stream: Perceptual and computational modeling of voice separation. *Music Perception*, 26(1):75–94. DOI: https://doi.org/10.1525/mp.2008.26.1.75

Chandna, P., Cuesta, H., and **Gómez, E.** (2020). A deep learning based analysis-synthesis framework for unison singing. In *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*, pages 598–604, Montreal, Canada (virtual).

Chew, E. and **Wu, X.** (2004). Separating voices in polyphonic music: A contig mapping approach. In *International Symposium on Computer Music Modeling and Retrieval*, pages 1–20. Springer. DOI: https://doi.org/10.1007/978-3-540-31807-1_1

Clift, S., Hancox, G., Morrison, I., Hess, B., Kreutz, G., and **Stewart, D.** (2010). Choral singing and psychological wellbeing: Quantitative and qualitative findings from English choirs in a cross-national survey. *Journal of Applied Arts & Health*, 1(1):19–34. DOI: https://doi.org/10.1386/jaah.1.1.19/1

Cuesta, H. (2022). *Data-driven Pitch Content Description of Choral Singing Recordings*. PhD thesis, Universitat Pompeu Fabra, Barcelona.

Cuesta, H., Gómez, E., and **Chandna, P.** (2019). A framework for multi-f0 modeling in SATB choir recordings. In *Proceedings of the Sound and Music Computing Conference (SMC)*, pages 447–453, Málaga, Spain.

Cuesta, H., Gómez, E., Martorell, A., and **Loáiciga, F.** (2018). Analysis of intonation in unison choir singing. In *Proceedings of the International Conference on Music Perception and Cognition (ICMPC)*, pages 125–130, Graz, Austria.

Cuesta, H., McFee, B., and **Gómez, E.** (2020). Multiple F0 estimation in vocal ensembles using convolutional neural networks. In *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*, pages 302–309, Montreal, Canada (virtual).

Cuthbert, M. S. and **Ariza, C.** (2010). Music21: A toolkit for computer-aided musicology and symbolic music data. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 637–642, Utrecht, The Netherlands.

Dai, J. and **Dixon, S.** (2019). Singing together: Pitch accuracy and interaction in unaccompanied unison and duet singing. *The Journal of the Acoustical Society of America*, 145(2):663–675. DOI: https://doi.org/10.1121/1.5087817

Devaney, J., Mandel, M. I., and **Fujinaga, I.** (2012). A study of intonation in three-part singing using the automatic music performance analysis and comparison toolkit (AMPACT). In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 511–516, Porto, Portugal.

Duan, Z., Han, J., and **Pardo, B.** (2013). Multi-pitch streaming of harmonic sound mixtures. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(1):138–150. DOI: https://doi.org/10.1109/TASLP.2013.2285484

Elsayed, N., Maida, A., and **Bayoumi, M.** (2019). Effects of different activation functions for unsupervised convolutional LSTM spatiotemporal learning. *Advances in Science, Technology and Engineering Systems Journal*, 4(2):260–269. DOI: https://doi.org/10.25046/aj040234

Gover, M. and **Depalle, P.** (2020). Score-informed source separation of choral music. In *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*, pages 231–239, Montreal, Canada (virtual).

Gray, P. and **Bunescu, R.** (2016). A neural greedy model for voice separation in symbolic music. In *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, pages 782–788, New York City, USA.

Hochreiter, S. and **Schmidhuber, J.** (1997). Long shortterm memory. *Neural Computation*, 9:1735–80. DOI: https://doi.org/10.1162/neco.1997.9.8.1735

Huron, D. (2001). Tone and voice: A derivation of the rules of voice-leading from perceptual principles. *Music Perception*, 19:1–64. DOI: https://doi.org/10.1525/mp.2001.19.1.1

Jin, Y. and **Wang, M.** (2020). LSTM model for single to dual track piano MIDI file. In *IEEE 9th Global Conference on Consumer Electronics (GCCE)*, pages 29–31, Las Vegas, USA. DOI: https://doi.org/10.1109/GCCE50665.2020.9291967

Jordanous, A. (2008). Voice separation in polyphonic music: A data-driven approach. In *Proceedings of the International Computer Music Conference (ICMC)*, Belfast, Ireland.

Kilian, J. and **Hoos, H. H.** (2002). Voice separation — a local optimisation approach. In *Proceedings of the 3rd International Conference on Music Information Retrieval*, pages 30–46, Paris, France.

Kingma, D. P. and **Ba, J.** (2014). Adam: A method for stochastic optimization. *ArXiv*, abs/1412.6980.

Kirlin, P. and **Utgoff, P.** (2005). VoiSe: Learning to segregate voices in explicit and implicit polyphony. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, pages 552–557, London, UK.

Kirsh, E. R., van Leer, E., Phero, H. J., Xie, C., and **Khosla, S.** (2013). Factors associated with singers' perceptions of choral singing well-being. *Journal of Voice*, 27(6):786–e25. DOI: https://doi.org/10.1016/j.jvoice.2013.06.004

Lordelo, C., Benetos, E., Dixon, S., and **Ahlbäck, S.** (2021). Pitch-informed instrument assignment using a deep convolutional network with multiple kernel shapes. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*.

Madsen, S. T. and **Widmer, G.** (2006). Separating voices in MIDI. In *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR)*, pages 57–60, Victoria, BC.

McLeod, A., Schramm, R., Steedman, M., and **Benetos, E.** (2017). Automatic transcription of polyphonic vocal music. *Applied Sciences*, 7(12). DOI: https://doi.org/10.3390/app7121285

McLeod, A. and **Steedman, M.** (2016). HMM-based voice separation of MIDI performance. *Journal of New Music Research*, 45:17–26. DOI: https://doi.org/10.1080/09298215.2015.1136650

Nakamura, E., Benetos, E., Yoshii, K., and **Dixon, S.** (2018). Towards complete polyphonic music transcription: Integrating multi-pitch detection and rhythm quantization. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 101–105, Calgary, Canada. DOI: https://doi.org/10.1109/ICASSP.2018.8461914

Petermann, D., Chandna, P., Cuesta, H., Bonada, J., and **Gómez, E.** (2020). Deep learning based source separation applied to choir ensembles. In *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*, pages 733–739, Montreal, Canada (virtual).

Raffel, C., McFee, B., Humphrey, E. J., Salamon, J., Nieto, O., Liang, D., and **Ellis, D. P.** (2014). mir_eval: A transparent implementation of common MIR metrics. In *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, pages 367–372, Taipei, Taiwan.

Rosenzweig, S., Cuesta, H., Weiß, C., Scherbaum, F., Gómez, E., and **Müller, M.** (2020). Dagstuhl ChoirSet: A multitrack dataset for MIR research on choral singing. *Transactions of the International Society for Music Information Retrieval (TISMIR)*, 3(1):98–110. DOI: https://doi.org/10.5334/tismir.48

Ryynänen, M. P. and **Klapuri, A. P.** (2008). Automatic transcription of melody, bass line, and chords in polyphonic music. *Computer Music Journal*, 32(3):72–86. DOI: https://doi.org/10.1162/comj.2008.32.3.72

Sarkar, S., Benetos, E., and **Sandler, M.** (2020). Choral music separation using time-domain neural networks. In *Proceedings of the DMRN+15:Digital Music Research NetworkWorkshop*, pages 7–8, London, UK.

Schramm, R. and **Benetos, E.** (2017). Automatic transcription of a cappella recordings from multiple singers. In *Proceedings of the AES Conference on Semantic Audio*, Erlangen, Germany.

Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K., and **Woo, W.-C.** (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Proceedings of the 28th International Conference on Neural Information Processing Systems (NeurIPS)*, pages 802–810.

Sigtia, S., Benetos, E., and **Dixon, S.** (2016). An endto- end neural network for polyphonic piano music transcription. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(5):927–939. DOI: https://doi.org/10.1109/TASLP.2016.2533858

Su, L., Chuang, T.-Y., and **Yang, Y.-H.** (2016). Exploiting frequency, periodicity and harmonicity using advanced time-frequency concentration techniques for multipitch estimation of choir and symphony. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 393–399, New York City, USA.

Tanaka, K., Nakatsuka, T., Nishikimi, R., Yoshii, K., and **Morishima, S.** (2020). Multi-instrument music transcription based on deep spherical clustering of spectrograms and pitchgrams. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 327–334, Montreal, Canada.

Weiß, C., Schlecht, S. J., Rosenzweig, S., and **Müller, M.** (2019). Towards measuring intonation quality of choir recordings: A case study on Bruckner's Locus Iste. In *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*, pages 276–283, Delft, The Netherlands.

TO CITE THIS ARTICLE:
Cuesta, H., and Gómez, E. (2022). Voice Assignment in Vocal Quartets Using Deep Learning Models Based on Pitch Salience. *Transactions of the International Society for Music Information Retrieval*, 5(1), 99–112. DOI: https://doi.org/10.5334/tismir.121

Submitted: 18 October 2021 **Accepted:** 03 March 2022 **Published:** 26 May 2022