©2018 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.

Appeared as: Bernhard Lehner, Jan Schlüter and Gerhard Widmer. Online, Loudness-invariant Vocal Detection in Mixed Music Signals. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 26(8):1369–1380, Aug. 2018.

http://doi.org/10.1109/TASLP.2018.2825108

Online, Loudness-invariant Vocal Detection in Mixed Music Signals

Bernhard Lehner, Jan Schlüter, and Gerhard Widmer

Index Terms—**singing voice detection, music information retrieval, neural network, lstm-rnn.**

***Abstract*—Singing Voice Detection, also referred to as Vocal Detection (VD), aims at automatically identifying the regions in a music recording where at least one person sings. It is highly challenging due to the timbral and expressive richness of the human singing voice, as well as the practically endless variety of interfering instrumental accompaniment. Additionally, certain instruments have an inherent risk of being misclassified as vocals due to similarities of the sound production system. In this paper, we present a machine learning approach that is based on our previous work for VD, which is specifically designed to deal with those challenging conditions.**

The contribution of this work is three-fold: First, we present a new method for VD that passes a compact set of features to an LSTM-RNN classifier that obtains state of the art results. Second, we thoroughly evaluate the proposed method along with related approaches to really probe the weaknesses of the methods. In order to allow for such a thorough evaluation, we make a curated collection of data sets available to the research community. Third, we focus on a specific problem that was not obvious and had not been discussed in the literature so far. The reason for this is precisely because limited evaluations had not revealed this as a problem: the lack of loudness invariance. We will discuss the implications of utilising loudness related features and show that our method successfully deals with this problem due to the specific set of features it uses.

I. Introduction

THE task of detecting human singing voice in mixed music signals – henceforth referred to as *vocal detection* (VD) – remains a challenging one.

Vocals could be considered a musical instrument, most likely the one with the highest amount of physical variation and emotional expressiveness. In consequence of a complicated movement of the jaw, tongue, and lips, the shape of the vocal tract is modified, thus enabling the singer to pronounce the lyrics of a song.

Furthermore, a modulation of the airflow through oscillation of the vocal folds allows to produce a wealth of different *timbres*[1] and a wide range of *fundamental frequencies* (f_0) of up to 10 octaves [1]. The perceived height of a note is independent of timbre and referred to as *pitch*, whereas f_0 is the main cue for pitch perception. Continuous pitch fluctuations were already used to detect singing voice e.g., in [2] and [3]. It seems they are a typical characteristic of vocals, but not exclusively. In Fig. 1, we can see the spectrograms of actual singing voice (upper left) along with examples of three instruments that are capable of producing similar sub-semitone pitch fluctuations. Therefore, instruments – especially those built to mimic the expressiveness of the human singing voice – have an inherent risk of being misclassified as vocals. We chose this specific voice example to demonstrate another fact about the human singing voice which is often overlooked. As we can see in the first four seconds of the spectrogram, it is also possible – at least for well trained singers – to hold the pitch perfectly. Consequently, the absence of pitch fluctuations does not imply the absence of vocals, and the presence of pitch fluctuations does not imply the presence of vocals.

Vocals and some instruments share not only the capability to produce pitch fluctuations, but also the capability to produce similar timbres. This is due to similarities in the sound production, e.g. a saxophone's reed resembles human vocal folds. Additionally, the practically endless variety of interfering instrumental accompaniment (see Fig. 2) contributes to the complexity of this task.

The very first attempt to tackle VD was done by Berenzweig and Ellis in [4], where they utilised the posterior probabilities of phonemes from a neural network based speech recogniser in order to derive a variety of models. After that, researchers often focused on engineering high-level features specifically for this task, or utilising features known from the speech processing domain, e.g. *Mel-Frequency Cepstral Coefficients* (MFCCs), *Linear Predictive Coefficients* (LPCs), *Perceptual Linear Predictive Coefficients* (PLPs).

Li and Wang [5] used a VD before they separated the vocals from instrumental accompaniment. They used MFCCs, LPCs, PLPs, and the 4-Hz harmonic coefficient as features, which they fed to a *Hidden Markov Model* (HMM) [6].

Ramona et al. used in [7] a very diverse set of features. These include MFCCs, LPCs, *zero crossing rate* (ZCR), sharpness, spread, f_0, and some aperiodicity measure based on the monophonic YIN library [8]. Furthermore, a multitude of features is extracted from two different time scales. All in all, their feature vector comprises 116 components, which is then reduced to 40 by a feature selection algorithm [9] and fed to a *Support Vector Machine* (SVM).

Mauch et al. [3] utilise four features in total, among them MFCCs. They introduce three novel features which are based on Goto's polyphonic f_0-estimator PreFEst [10]: *Pitch fluctuation*, which is basically the standard deviation of intra-semitone f_0 differences. In addition to the MFCCs of the untouched signal, the authors also propose *MFCCs of the re-synthesised predominant voice*, and *normalised amplitude of harmonic partials*. An SVM-HMM [11], [12] is used for classification.

BL and GW are with the Department of Computational Perception, Johannes Kepler University, Linz, Austria (e-mail: bernhard.lehner@jku.at;gerhard.widmer@jku.at); JS is with the Austrian Research Institute for Artificial Intelligence, Vienna, Austria (e-mail: jan.schlueter@ofai.at).

[1]The distinguishable particular quality of a sound

Weninger et al. [13] extracted a 46-dimensional feature vector containing the first 13 MFCCs (including the 0^{th}), along with their first and second order derivatives (delta and double delta), short-time energy, its zero- and mean-crossing rate, voicing probability, f_0, harmonics-to-noise ratio, and predominant pitch computed with the open-source toolkit openSMILE [14]. The features are extracted after applying a source separation algorithm specifically designed to extract the vocals of the lead singer. As a classifier, they use *Bidirectional Long Short-Term Memory Recurrent Neural Networks (BLSTM-RNNs)*, which have access to the complete past and future context. Originally, this method was developed in order to identify the gender of the lead singer, but they also report excellent results for the VD task.

In [15], Hsu et al. used *Gaussian Mixture Models* (GMMs) as states in a fully connected HMM with the Viterbi algorithm [6]. They used *Harmonic/Percussive Source Separation* (HPSS) as a pre-processing step. Their 39-dimensional feature vector contains 12 MFCCs, the log energy, and their first and second order derivatives.

In [16], it was shown that only appropriately selected and optimised MFCCs fed to a Random Forest classifier could achieve recognition results that are almost on par with more complicated methods.

The current state of the art with a feature engineering approach is proposed in [17], where features like MFCCs, Fluctogram [18], and some reliability indicators are fed to an LSTM-RNN. The method is real-time capable, has a low latency, and was evaluated on several data sets, most of them publicly available.

Recently, researchers also started utilising feature learning from a low-level representation with deep learning methods. Leglaive et al. fed pre-processed mel-scaled spectrograms (by a two stage HPSS) to deep BLSTMs in [19]. They iteratively extended the architecture by hidden layers based on results on the test set of the Jamendo corpus [7], and report no evaluation results on truly unseen data.

The current state of the art utilising *Convolutional Neural Networks* (CNNs) on mel spectrograms was proposed by Schlüter and Grill in [20] and further refined in [21, Sec. 9.8]. They apply data augmentation techniques (pitch shifting, time stretching, and frequency filters) in order to improve performance. Without data augmentation – which most likely improves other methods as well – the performance seems to be on par with the feature engineering approach from [17].

The specific contributions of this paper are three-fold: As our first contribution, we describe a compact, light-weight method for VD that further improves upon previous work of ours [17] in Section II. Despite similarities of the classifier and feature extraction, we managed to reduce the computational burden, remove a weakness related to varying levels of loudness, reduce the tendency of the algorithm to misclassify certain instruments as vocals, while still improving the overall performance.

Naturally, such claims cannot be made without proper assessment over multiple and diverse sets of data. We consider the inclusion of instrumental music and evaluation across data sets of paramount importance. In doing so, we increase the relevance of the results. Unfortunately, openly available data sets to evaluate VD methods are scarce. Therefore, curated data sets based on previous findings (high risk of false positives with certain instruments) are made available to the research community: Ground truth annotations for one openly available data set containing 100 songs, and six smaller data sets containing instrumental music that were collected and sorted into specific categories relating to the predominant instruments. Those and other publicly available data sets will be explained in more detail in Section III. Our second contribution is the evaluation procedure in conjunction with that data, and will be discussed in Section IV.

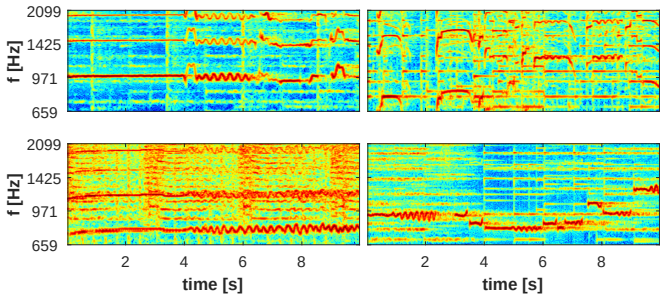

Fig. 1. Spectograms of instruments capable of producing voice-like pitch trajectories. Upper left plot: actual singing voice; upper right: saxophone; lower left: electric guitar; lower right: pan flute.

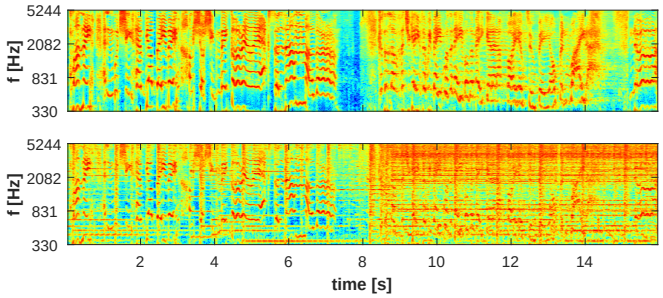

Fig. 2. Spectograms to demonstrate interferences of instrumental accompaniment. Upper plot: vocals only; lower plot: mixed version. Especially in the second half the interferences are quite severe, making it hard to extract information that relates solely to vocals.

In Section V, we will expose that the previous evaluation procedure still yields limited insights. As our third contribution, we focus on a specific problem that had not been discussed in the literature so far: the lack of *loudness invariance*. We will discuss why utilising loudness related features makes the outcome of a standard evaluation procedure less meaningful. This paves the way to recognise the necessity of a different evaluation strategy, which we then propose. We finally demonstrate with concrete examples that even though for some data sets the accuracy is equal for two methods, they behave quite differently when we evaluate loudness invariance according to the proposed evaluation.

II. Method

In this section, we discuss a set of features that is specifically tailored for the task of VD in combination with LSTM-RNNs. This is the first contribution of our work, and a

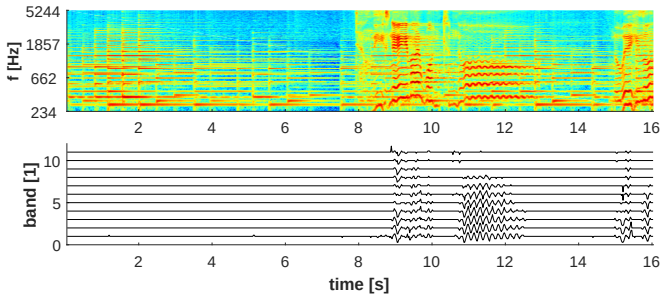

Fig. 3. The Fluctogram (lower plot) targets only sub-semitone fluctuations in the second half of the spectrogram (upper plot), not the discrete pitch changes in the first half.

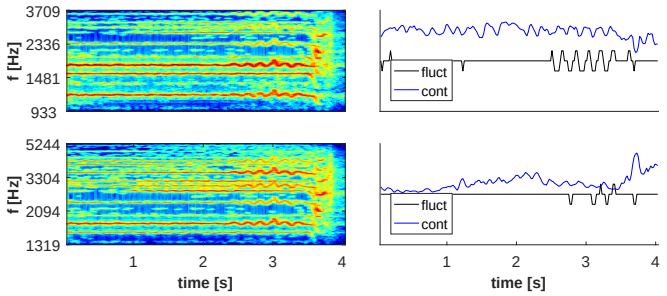

Fig. 4. Left side: spectrograms representing the 9th band (upper plot) and the 11th (lower plot). Right side: the corresponding Fluctogram (fluct) and Spectral Contraction (cont). The Fluctogram is most reliable when a large amount of energy is located near the center (notice how the vibrato at the end is only well captured in the upper plot). In such cases, the higher reliability is indicated by a higher Spectral Contraction.

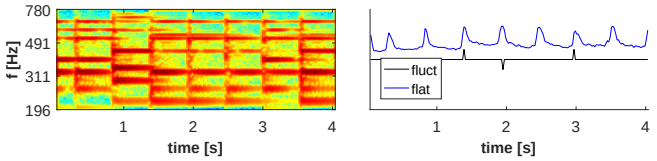

Fig. 5. Left side: spectrogram of piano onsets. Right side: the corresponding Fluctogram (fluct) and Spectral Flatness (flat). As can be seen, the percussive nature of those onsets - even though they stem from a harmonic, pitch-discrete instrument - causes occasionally false positive pitch fluctuations. A high Spectral Flatness (notice the increased values corresponding to the onsets) indicates low reliability of the Fluctogram in such cases, enabling us to ignore those non-existing pitch fluctuations.

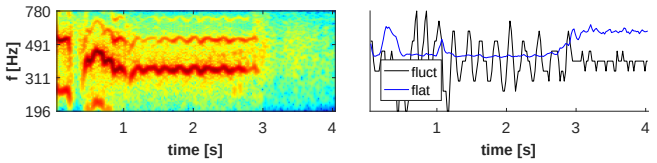

Fig. 6. Left side: spectrogram of actual singing. Right side: the corresponding Fluctogram (fluct) and Spectral Flatness (flat). The quiet last second causes some false positive pitch fluctuations. Again, a high Spectral Flatness (notice the increased value in the last second) indicates low reliability of the Fluctogram in such cases.

continuation of previous research described in [17]. Regarding our set of features, it is worth mentioning that all of them are completely invariant to the level of energy/loudness – a design choice that leads to a desirable robustness which will be discussed later in Section V. The measurements taken to improve upon our previous method all contribute approximately equally and are as follows. Contrary to our previous approach, the reliability indicators are not fed to the classifier anymore, but used to post-process the Fluctogram. By using only a compact set of features, we drastically reduce the number of weights in the LSTM-RNN. This procedure limits the network's capability to fit the training data and prevents overfitting, hence acting as a regulariser. As a consequence of these modifications, our method is less prone to false positives, which will be demonstrated later in Section IV.

A. Classifier

RNNs are neural networks designed for stepwise processing of sequential data, equipped with feedback (recurrent) connections to access the previous step's internal state. This allows RNNs to keep information in memory, and model an indefinite temporal context. During training, recurrent connections can lead to vanishing or exploding gradients, which Hochreiter and Schmidhuber proposed to mitigate with *Long Short Term Memory* (LSTM) units in [22]. The RNNs' capability to learn the amount of temporal context needed for classifying the current frame can be an advantage over approaches with a fixed-size context such as HMMs or CNNs.

LSTM-RNNs proved to be very successful and have delivered state of the art performance in a wide range of tasks where the temporal context of a signal is important, e.g. in handwriting recognition [23] or phoneme recognition [24]. Since temporal context is sometimes also necessary for humans to make a *vocal-nonvocal* decision,[2] it seems natural to use LSTM-RNNs for VD.

B. MFCCs

The spectral envelope of an audio signal is strongly related to timbre, and envelope descriptors like LPCs, PLPs, or MFCCs are used in most state of the art VD methods. Among the aforementioned descriptors, MFCCs [25] are the most widely used audio features, especially for *Music Information Retrieval* (MIR) tasks. In [16], it was shown that it can make a substantial difference if MFCCs are parametrised towards a specific task, in this case VD. Classification results using only such optimised MFCCs along with their first order derivatives (deltas) seemed to be on par with sometimes more complicated state of the art methods.

In several experiments with our internal train data set, we discovered that for classifiers like Random Forests or SVMs as used in [16], [18], [26], [27], MFCCs turned out to be the most useful features, and adding their deltas increased the performance only slightly. However, it seems to be different when using sequential classifiers like RNNs as in [13], [17]. In this case, our experiments revealed quite a different ranking of feature importance. The MFCC deltas turned out to be the most useful features, while adding MFCCs themselves even lowered the performance. Regardless of the classifier, MFCC double deltas never turned out to be useful. Therefore, we only

[2]examples at `http://www.cp.jku.at/misc/ieee2017vd/`

use the deltas of MFCCs $0-17$, resulting in 18 attributes in our proposed method.

The observation window to compute the MFCCs is 100 ms long, and always placed symmetrically around the current frame of 20 ms, that is, we use additional 40 ms of the signal in both directions.

C. Fluctogram and Reliability Indicators

The Fluctogram is a refinement of a feature that was initially introduced by Sonnleitner et al. [28] to detect the presence of speech in mixed audio signals. This feature was based on the observation that spectrograms of speech signals tend to display patterns of tonal components (i.e. partials) that vary in frequency over time. Since vocals often exhibit a similar characteristic, we take the basic idea of computing the *cross-correlation* of neighbouring frames.

In particular, each magnitude spectrum of a time frame $|X_n|$ is compared to the subsequent one $|X_{n+1}|$ by computing the cross-correlation. The actual feature value is the index of the maximum correlation when $|X_{n+1}|$ is shifted $\pm m$ frequency bins. A non-zero feature value indicates a pitch fluctuation. While we could instead employ f_0-estimation and compute the temporal difference between f_0 estimates, directly cross-correlating spectral frames has two advantages: First, multiple pitch estimation is still considered an open problem for mixed musical signals, and potential errors will be propagated through the remaining processing chain. Second, vocals are not always predominant, therefore characterising predominant pitch trajectories would give results that are not always targeted at vocals.

1) Fluctogram: The procedure to compute the Fluctogram is as follows. First, we perform a *Discrete Fourier Transform* (DFT) on 20 ms audio frames to obtain the short-term magnitude spectrum $|X_n[f]|$. The actual observation window to compute the spectrum is 100 ms long, and always placed symmetrically around the current frame, that is, we use additional 40 ms of the signal in both directions. In order to ensure proper frequency resolution in the lower region for the logarithmic scaling, we apply a zero padding factor of 2^3.

We then map the frequency axis of the spectrum to a logarithmic scale that relates to pitch. The rationale behind this is that fluctuating trajectories of the partials need to be equidistant for the cross-correlation to reveal them. We suggest a pitch scale that spans 4.5 octaves from A#3 (233 Hz) to E8 (5274 Hz). The result is a logarithmically scaled spectrum with 540 bins, where 10 bins cover the range of one semitone. Notice that the pitch of vocals could well be beneath the lower boundary of this scale. Nevertheless, it is very likely to capture pitch fluctuations, since the relatively high amount of partials that are produced along with low pitched sounds still influences the result of the cross-correlation in the region above the actual pitch.

Afterwards, we divide this spectrum into 11 overlapping bands, each band 240 bins wide, spanning two octaves. The bands are always 30 bins apart from the next, which equals three semitones. Intuitively, it would make sense to set the individual bands only one semitone apart, but experiments with our internal data led to the conclusion that this mostly just increases the size of the feature vector without significantly increasing the performance.

We then weight each band by a triangle window that matches its bandwidth to reduce the influence of partials that could cross the band boundaries.

The harmonic fluctuations within each band are then revealed by pinpointing the maximum cross correlation at shifts of ± 5 bins, which equals half a semitone. Therefore, only sub-semitone, pitch-continuous fluctuations are targeted and detected, as can be seen in Fig. 3. Notice that we don't capture the actual trajectories of the harmonic fluctuations depicted in the spectrogram, but their first order derivatives.

2) Reliability Indicators: As will be demonstrated later on, the Fluctogram is error-prone under certain circumstances, and some fluctuations are either not present in the signal at all, or not well captured. To alleviate this, additional information that characterises the reliability of the information captured in the individual bands of the Fluctogram is needed. In short, the Fluctogram is most reliable when most of the energy is concentrated near the center of the frequency band, and the signal is more harmonic and less like white noise. Therefore, we suggest to compute two additional descriptors that we use to post-process the Fluctogram.

The first reliability indicator, which we call *Spectral Contraction (SC)* [18], was inspired by *Spectral Dispersion* (SD) [29]:

$$sd[n] = \sum_{j=0}^{N-1} |X_n[j]|^2 |j - f_c|, \tag{1}$$

where n is the index of the frame, N the number of bins of the spectrum, X_n the spectrum, j the index of a bin, and f_c the index of the central bin of the spectrum.

Basically, SD indicates how much of the energy resides in the center of the power spectrum $|X_n|^2$, and the smaller its value, the more energy is concentrated near the center. The fact that SD was not developed with our specific use-case in mind contributes to two problems. First, the result is energy-dependent (unless derivatives are used as actually suggested for beat-tracking in [29]). Second, the applied weighting $|j - f_c|$ of the power spectrum is effectively an inverse triangle window, which we consider too sensitive to pitch fluctuations.

With that in mind, we suggest to use the ratio of the weighted power spectrum $|X_n|^2 * w$ to the power spectrum $|X_n|^2$ itself to compute SC, as given in Equation 2. The result is loudness invariant and always in the range $[0-1]$, where small values indicate that the energy is widely dispersed. Large values indicate that the energy is primarily concentrated near the center. In order to reduce the sensitivity towards sub-semitone pitch fluctuations, we suggest as weighting window w a Chebyshev window that matches the bandwidth with a sidelobe attenuation of 200 dB.

$$sc[n] = \frac{\sum_{j=0}^{N-1} |X_n[j]|^2 w[j]}{\sum_{j=0}^{N-1} |X_n[j]|^2} \tag{2}$$

In Fig. 4 we demonstrate the usefulness of the SC feature, where we can see the spectrograms of the same signal, but from two different frequency bands. The upper left plot depicts the spectrogram of the 9th band, and the lower left depicts the spectrogram of the 11th frequency band. On the right sides we can see the corresponding Fluctogram and SC values. The vibrato towards the end is well captured by the Fluctogram in the upper right plot, but not so much in the lower right plot. By comparing the spectrograms, a correlation between the amount of energy near the center and the reliability of the Fluctogram reveals itself: The more the energy is concentrated near the center, the more reliable the Fluctogram becomes. In order to keep only well captured fluctuations, we reset Fluctogram values to 0 where the corresponding SC is below 0.1. We don't actually feed this feature to the classifier directly.

The second reliability indicator describes the similarity of the signal to white noise, and we suggest the *Spectral Flatness (SF)* measure [30]. It is usually computed as the log-scaled ratio of geometric mean to arithmetic mean of the power spectrum. However, for the sake of simplicity, we suggest to compute SF as follows:

$$sf[n] = \frac{\sqrt[N]{\prod_{j=0}^{N-1} |X_n[j]|}}{\frac{1}{N}\sum_{j=0}^{N-1} |X_n[j]|} \tag{3}$$

The result is loudness invariant and always in the range $[0-1]$, where small values indicate high harmonicity and high values indicate a high similarity to white noise.

Again, we don't feed this feature to the classifier, but we use it to reset Fluctogram values to 0, as soon as the SF exceeds a value of 0.6. The justification for this measurement is given in Fig. 5, where we can see the occasional false positive fluctuations corresponding to percussive onsets of an otherwise harmonic instrument (piano). Another example of false positive fluctuations is given in Fig. 6, caused by a high amount of noise towards the end of the plot.

The resulting Fluctogram does not suffer from false positives or poorly captured fluctuations anymore, and has 11 feature values.

D. Complete Feature Set and Final Classifier

For VD, the audio signal is analysed and classified with a time resolution of 20 ms. That is, we have 50 training examples per second, each characterised by 29 features computed around the current time point: 11 Fluctogram features, post-processed by two different reliability indicators, and 18 MFCC-based features (deltas from MFCCs 0-17).

All of the features can be calculated from the same spectrogram with an observation window length of 100 ms, centered around a 20 ms frame. After extraction, we standardise features to zero-mean and unit variance according to the train set. The validation and test sets are always left unseen in this regard, and normalised according to the train set.

The LSTM-RNN has one input layer matching the size of the feature vector (29), one hidden layer with 55 LSTM units, and one softmax layer with two units.[3] The weights are randomly initialised from a zero-mean Gaussian distribution with a standard deviation of σ=0.05. We add noise to the input features for improved generalisation in the form of another zero-mean Gaussian distribution (σ=0.3).

Each song from the train set is then presented to the LSTM-RNN on a frame-by-frame basis in correct order. The weights are updated with a steepest descent optimiser [31] (rate=10^{-5}, momentum=0.9) in order to minimise the cross entropy error [31].

III. Data

In this section, we stress the importance of proper data sets for VD evaluation based on a discussion of three major problems/threats: first, the threat to produce false positives by mistaking instruments as vocals; second, the risk of missing vocals with low SNR; third, the danger of getting overly optimistic results due to something that we would call the *data set effect*. As a consequence, we publish novel data sets, each carefully designed to reveal the three biggest potential weaknesses of VD algorithms.

A. Threat 1: False Positives

As previously discussed, highly harmonic instruments have an increased risk for being misclassified as vocals. Usually, metrics like the false positive rate or precision on songs (i.e., recordings that actually contain a singing voice) are used to estimate the performance in this regard. However, a high precision does not necessarily reflect a high robustness against false positives. Only the presence of a large number of adversarial examples allows for a meaningful robustness evaluation. Therefore, we suggest to incorporate instrumental music in the evaluation, preferably instrumental cover versions, where the vocalist is replaced by a highly expressive instrument. For the remainder of this paper, we refer to such recordings as *instrumentals*, and to recordings actually containing vocals as *songs*. Fortunately, it is relatively easy to compile a data set which contains solely instrumentals. In the past [18], we already identified instruments that tend to produce false positives (strings, electric guitars, flutes, and saxophones), and we just need to compile instrumentals having those as lead instruments. Every frame in an instrumental that is classified as *vocal* by a VD algorithm is a false positive then.

Since different instruments challenge VD algorithms in different ways, we suggest to analyse them separately.

B. Threat 2: Low SNR

Many data sets used in the literature contain professionally recorded and mastered songs. Usually, those recordings have relatively high SNRs, which in our case could also be described as *vocals-to-accompaniment ratio*. However, since the advent of digital music distribution services like Jamendo [32], many recordings are available that are produced with less-than-optimal recording equipment and mixing/mastering skills, often with a low SNR. Therefore, we suggest to evaluate VD algorithms also in this regard.

[3]The softmax output layer is not necessary for a 2-class problem, but can easily be modified to support N-class problems.

C. Threat 3: Data Set Effect

The data set effect occurs for many reasons and is not so easy to spot. It is closely related to overfitting, and considers also the possibility that overfitting is present but not noticeable due to data set specifics. In general, the data set effect comprises all the reasons that could lead to overly optimistic results, like similarities of training and testing data in terms of audio codec, genre, artist, recording or mastering equipment, instrument and vocal timbre, mood, rhythm, loudness, singing style, and vocalist gender. Related to this are the *artist* and *album effects* discussed in the Music Information Retrieval (MIR) literature, mostly in the context of audio similarity [33] and artist identification [34].

Since it sometimes occurs that research is done on data sets without in-depth knowledge of such subtle relationships, one could end up totally unaware of the presence of this effect. However, by using different data sets for training and testing respectively, this pitfall can be easily avoided, yielding more realistic results.

To give an example, for the RWCMDB-P-2001 data set [35] it is common practice to report 5-fold CV results where the folds are randomly generated [3], [16]–[18], [20]. Due to production resource constraints, the 100 songs are performed by only 34 singers. Obviously, by randomly dividing such a data set, some singers will end up in the training as well as in the test set, rendering the results less meaningful. Therefore, we suggest to use this data set exclusively for testing. An overview of our train, validation, and test setup is given in Table I, and will be discussed in more detail in the following sections.

D. Train Data

The train data is composed of 316 audio recordings, for a total of 20 hours, 13 minutes. Approximately 30% of the frames are annotated as vocal, and the amount of *a capella* singing, i.e. without instrumental accompaniment, is negligible. As already stated, we would prefer to use complete data sets in either train, validation, or test set. However, we kept the original split of the jamendo [32] data set, and added the subsets to our own collection accordingly. This was necessary to ensure a sufficient amount of vocal examples in the training phase. The train data contains the following data sets:

- *golden pan* (pan flute instrumentals)
- *golden sax* (saxophone instrumentals)
- *rockband* (rock songs)
- *jamendo* training (pop/rock songs)
- *heavy instr.* (electric guitar instrumentals)
- *opera* (opera arias)

The opera songs comprise a selection from the operas *La Traviata*, *Madame Butterfly*, and *Die Zauberflöte*.

E. Validation Data

Validation data is used for early stopping [31] (no improvement after 20 epochs) in order to yield models with good generalisation capabilities. Additionally, the results on the validation data were used to find suitable thresholds for resetting non-reliable Fluctogram values as previously discussed in Section II-C2. It is composed of 239 audio recordings, for a total of 17 hours, 38 minutes. Approximately 21% of the frames are annotated as vocal. The validation data contains the following data sets:

- *dg* (opera songs from Don Giovanni)
- *hi* (electric guitar instrumentals)
- *jamendo* validation (pop/rock songs)
- *pakarina* (pan flute instrumentals)
- *rb* (rock songs)
- *softj* (saxophone instrumentals)
- *sq* (string quartet instrumentals)

F. Test Data

The test data contains only audio recordings available to the research community, and is unseen in all regards. It is composed of 545 audio recordings, for a total of 41 hours, 48 minutes. Approximately 21% of the frames are annotated as vocal. The following data sets are not provided by the authors, but available for research from public sources.

- *jamendo* test (pop/rock songs)
- *rwc_pop* (pop/rock songs)
- *rwc_classical* (orchestra instrumentals)
- *rwc_jazz* (jazz instrumentals)

Data sets with the prefix *rwc_* are all part of the RWC Music Database [35]. We had to ignore two recordings from the *rwc_jazz* data set, since they contained singing voice. The exact list of recordings that we used for our evaluation is available on the accompanying web page to this article. For the data set *rwc_pop* we revised the ground truth annotations, which we will also release.

The following data sets are part of our second contribution, and are available online as well. They are organised in different categories mainly with respect to the most challenging instruments for VD algorithms known to us so far.

- *msd100* (rock songs)
- *yt_classics_song* (opera songs)
- *yt_classics_instr* (orchestra instrumentals)
- *yt_guitars* (acoustic guitar instrumentals)
- *yt_heavy_instr* (electric guitar instrumentals)
- *yt_wind_flute* (flute instrumentals)
- *yt_wind_sax* (saxophone instrumentals)

Specifically with the threat of low SNR in mind, we propose to use the publicly available Mixing Secret Dataset 100 (msd100) [36], for which we manually prepared annotations for the presence of vocals. This data set was initially used for source separation evaluation, and includes 4 stereo sources corresponding to the bass, the drums, the vocals and the remaining instruments for each of the 100 songs. Although the songs are professionally recorded, the provided mixes are not professionally mixed and mastered (at least according to our own judgement). Some songs contain vocals that are hard to perceive, even for human listeners, which makes it a very challenging data set that helps to scrutinise VD algorithms.

Data sets with the prefix *yt_* were selected from Youtube, and will be provided as file lists. All ground truth annotations will be made available directly.

TABLE I
OVERVIEW OF THE DATA SETS, FOR EACH SET INDICATING IF IT ACTUALLY CONTAINS SINGING VOICE, THE AVAILABILITY, IF WE CURATED IT BY OURSELVES, AND IF WE CONSIDER IT SUITED TO HELP (WHEN TRAINING) OR GIVE INSIGHT (WHEN TESTING) REGARDING THE AFOREMENTIONED THREATS.

data set	contains vocals	publicly available	annotated ourselves	# recordings	length [min]	threat 1 (false pos.)	threat 2 (low snr)	threat 3 (data set effect)
Train set								
jamendo train	✓	✓	–	61	239	✓	–	–
opera	✓	–	✓	32	142	–	–	✓
rockband	✓	–	✓	75	296	✓	–	–
golden pan	–	✓	✓	28	103	✓	–	✓
golden sax	–	✓	✓	100	345	✓	–	✓
heavy instrumentals	–	–	✓	20	88	✓	–	–
all	**–**	**–**	**–**	**316**	**1213**	**–**	**–**	**–**
Validation set								
dg	✓	–	✓	13	45	–	–	✓
jamendo validation	✓	✓	–	16	61	✓	–	–
rb	✓	–	✓	74	299	✓	–	–
hi	–	–	✓	72	345	✓	–	–
pakarina	–	✓	✓	15	70	✓	–	✓
softj	–	✓	✓	29	152	✓	–	✓
sq	–	✓	✓	20	68	✓	–	✓
all	**–**	**–**	**–**	**239**	**1040**	**–**	**–**	**–**
Test set								
jamendo test	✓	✓	–	16	71	–	–	–
msd100	✓	✓	✓	100	417	–	✓	✓
rwc_pop	✓	✓	✓ (revised)	100	407	–	–	✓
yt_classics_song	✓	✓	✓	9	47	–	–	✓
rwc_classical	–	✓	✓	55	308	✓	–	✓
rwc_jazz	–	✓	✓	48	218	✓	–	✓
yt_classics_instr	–	✓	✓	112	622	✓	–	✓
yt_guitars	–	✓	✓	12	51	✓	–	✓
yt_heavy_instr	–	✓	✓	50	209	✓	–	✓
yt_wind_flute	–	✓	✓	21	76	✓	–	✓
yt_wind_sax	–	✓	✓	22	82	✓	–	✓
all	**–**	**–**	**–**	**545**	**2508**	**–**	**–**	**–**

With these data sets, we can now evaluate VD algorithms in more detail compared to what has been done in the literature so far. In the following Section we will demonstrate that even though some methods are conceptually very close (all LSTM-RNNs with MFCCs+more features), on some data sets they are quite different in behaviour.

IV. EXPERIMENTS

In the following section, we present the results of several experiments. As feature engineering baselines for comparison we chose the methods from Weninger et al. [37][4] and our previous approach from [17] that we already briefly introduced in Section I. This selection is based on the fact that they all are conceptually very close to our proposed approach: MFCCs and some additional features fed to an LSTM-RNN classifier.[5] As a feature learning baseline we chose the method from Schlüter [21, Sec. 9.8]. In order to investigate the impact of data augmentation (pitch shifting up to $\pm 30\%$, time stretching up to $\pm 30\%$, and frequency band filters up to ± 10 dB; deemed the optimal combination in [20]), we report results achieved with and without it.

For the remainder of this paper we refer to the methods as follows. **WEN**: Weninger et al. [37]; **LEH**: Lehner et al. [17]; **CN1, CN2**: Schlüter [21, Sec. 9.8] without and with data augmentation, respectively;[6] **NEW**: the proposed method. All results were achieved with a single model each (i.e., no ensembling), selected from 24 models trained with different random initialisations for each method. The model selection for each method was based on the best performance according to the results on the validation set.

The key aspects of all four methods are listed in Table II. Notice that our proposed method NEW has the least number of features and learnable network parameters (i.e., weights). The minimum latency due to the feature extraction is explained in detail in [17]. All methods are online capable, except method WEN due to the use of a BLSTM-RNN. This variant of the LSTM-RNN requires access to the complete future context of the sequence, hence turning method WEN into an offline method.

In Table III the results of validation and test sets are listed in terms of accuracy. For the remainder of this section, we

[4]We do not utilise the source separation from [37] as this will most likely improve the other methods as well, according to [27]

[5]For all methods, we utilise *RNNLIB* from Alex Graves [38]

[6]Specifically, the baseline from https://github.com/f0k/ismir2015/tree/phd_extra, with an adapted learning rate schedule to account for the larger data set, dropping the rate when the error plateaus and stopping on the third plateau.

TABLE II
OVERVIEW OF THE KEY ASPECTS OF THE METHODS.

	WEN	LEH	CN1/CN2	NEW
# features	46	111	n/a	29
# weights	82 k	37 k	1600 k	19 k
min. latency [ms]	200	140	1260	140
online capable	no	yes	yes	yes
loudness invariant	no	no	no	yes

TABLE III
VALIDATION AND TEST SET RESULTS (ACCURACIES [%]). THE UPPER SECTIONS OF EACH OF THE TWO TABLES CONTAIN THE RESULTS REGARDING ACTUAL SONGS, AND THE LOWER SECTIONS RELATE TO PURE INSTRUMENTAL MUSIC.

Validation Set					
data set	**WEN**	**LEH**	**CN1**	**CN2**	**NEW**
dg	89.2	90.2	91.9	92.4	90.2
jamendo	80.1	80.8	88.8	91.3	84.0
rb	85.2	86.0	91.1	92.4	88.8
all song	**84.9**	**85.7**	**90.8**	**92.2**	**88.2**
hi	98.5	98.4	99.2	99.1	99.8
pakarina	94.8	97.6	98.3	98.9	99.8
softj	98.1	99.5	99.1	99.7	99.4
sq	92.0	99.9	99.1	98.8	99.6
all instr	**97.3**	**98.7**	**99.1**	**99.2**	**99.7**
all validation	**92.5**	**93.7**	**95.9**	**96.5**	**95.2**

Test Set					
data set	**WEN**	**LEH**	**CN1**	**CN2**	**NEW**
jamendo	84.2	87.0	92.1	93.2	85.1
msd100	70.8	75.7	78.4	79.9	78.4
rwc_pop	77.5	87.7	85.4	90.6	87.7
yt_classics_song	84.9	90.0	91.6	93.7	89.0
all song	**75.4**	**82.4**	**83.1**	**86.2**	**83.5**
rwc_classical	97.2	99.9	99.9	99.9	99.9
rwc_jazz	97.0	98.7	99.7	99.9	99.8
yt_classics_instr	98.9	99.8	99.3	99.4	100
yt_guitars	99.1	99.5	99.3	99.5	100
yt_heavy_instr	98.5	97.1	98.4	99.4	99.8
yt_wind_flute	98.4	96.4	99.2	99.6	99.4
yt_wind_sax	82.2	96.6	99.1	98.6	99.3
all instr	**97.4**	**98.9**	**99.3**	**99.5**	**99.5**
all test	**89.1**	**92.8**	**93.2**	**94.5**	**93.5**

focus on the test set results – the only results on truly unseen data.

Regarding songs (row **all song**), our proposed method NEW outperforms method WEN by 8.1 *percentage points* (ppt), method LEH by 1.1 ppt, method CN1 by 0.4, and is only outperformed by method CN2 by 2.7 ppt. The data set *msd100* seems to be the most challenging. This is not surprising, since we specifically selected it in order to evaluate performance regarding low SNR of vocals (Threat 2). All methods suffer from a low recall (i.e., true positive rate), and method NEW is on par with CN1, and outperforms method WEN by 7.6 ppt, and method LEH by 2.7 ppt. CN2 outperforms method NEW by 1.5 ppt.

Regarding instrumentals (row **all instr**), our proposed method NEW is on par with method CN2, outperforms method WEN by 2.1 ppt, and method LEH by 0.6 ppt, and method CN1 by 0.2 ppt. The data set *yt_wind_sax* leads to a relatively high number of false positives in general, but seems to be specifically challenging for method WEN, which produces 17.8% false positives.

If we would try to interpret the test set results, it would be plausible – hence tempting – to draw, inter alia, the following conclusions: (1) Overall, CN1 is on par with NEW; (2) CN2 performs better on songs, and on par on instrumental music compared to NEW; (3) WEN handles all instruments relatively well, except saxophone music; (4) LEH seems to have just a slight weakness on wind instruments. In order to support those conclusions even more, we could also report results based on other metrics like precision, recall, and f-measure. Furthermore, statistical significance tests if the best performing method is an actual improvement over the other approaches seem to be appropriate.

However, we do not report any more results in order to avoid the incorrect impression that they would improve the quality of the evaluation. This is based on the realisation of the severe implications on the interpretability of standard evaluation results caused by a lack of loudness-invariance. As we will demonstrate in the following section, an evaluation that disregards loudness-invariance yields misleading results – regardless of the evaluation metrics.

V. LOUDNESS

In this section, we will demonstrate the negative effects of utilising loudness-related features like 0^{th} MFCC, as is done in the baseline methods WEN [37] and LEH [17]. Furthermore, we will demonstrate that this is also an issue for our feature learning baseline methods CN1 and CN2 [21, Sec. 9.8], and data augmentation does not lead to satisfactory results. In this work, *loudness* refers to some strictly increasing function of the signal power, such as the 0^{th} MFCC. The exact definition is not important because we are only interested in loudness *invariance*. Since explicit information about loudness invariance is usually not provided, we propose an evaluation strategy that specifically targets the negative impact on performance caused by varying levels of loudness.

After investigating the Jamendo train set [32], an almost perfect linear correlation revealed itself: the higher the value of the 0^{th} MFCC, the higher the probability of the frame being annotated as *vocal*. We utilised the 0^{th} MFCC also in our previous works [16]–[18], [26], [27], since it led to a raise in accuracy of approximately 4 ppt on an internal data set comprising only songs. However, by including instrumental music in the evaluation, we discovered that the utilisation of features correlated with loudness increases the risk of instrument-vocal misclassification. After further investigations, it became clear that the negative impact is more severe than initially estimated, since we could also trick models into generating false positives on non-musical sound sources. The details of this will be discussed in the next section.

A. Adversarial Examples

In this section, we will demonstrate that loudness-related features will end up having a high importance for the prediction, even though the model was trained with both songs and instrumental music.

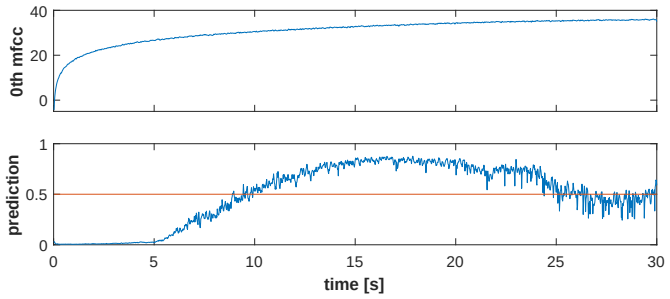

Fig. 7. Just white noise with increasing loudness (reflected by 0^{th} MFCC in the upper plot) is all it takes to generate false positives ($\sim 50\%$ of the predictions in the lower plot are above threshold), even though the model (WEN) performed relatively well on the experiments in the previous Section.

First, we give an example of a *rubbish adversarial*, that is, we demonstrate that we can induce misclassification on white noise – a signal that is clearly non-musical. For that, we generate 30 seconds of white noise, and linearly increase the volume from the beginning to the end. Fig. 7 shows in the upper plot the corresponding values of the 0^{th} MFCC, and in the lower plot the predictions of the model from method WEN. Reaching probabilities up to 0.9, the model predicts approximately 50% of this example as *vocal*. On recordings from the test set, we could also observe that the same model produces false positives on applause, again something clearly non-musical.

Another kind of adversarial examples are those that are perceptually almost indistinguishable from the original, but lead to an erroneous output. Changing the loudness of an audio recording is a modification where the result is perceptually very close to the original recording. However, for algorithms that incorporate loudness-related information, this modification can have a severe impact on their performance.

Fig. 8 shows the effect on the behaviour of the model from method WEN on two examples of instrumental music taken from the test set. Along with spectrograms, there are three posterior probability plots, each stemming from either increased, untouched, or decreased loudness. We can make two interesting observations: first, just changing the loudness slightly by ± 3 dB can increase the amount of false positives considerably; second, there is no common weakness regarding a specific level of loudness modification: in the upper example it is a decreased loudness that causes more false positives, and in the example below an increased loudness. It seems that for every recording there is a different level of loudness that triggers the highest numbers of errors. This raises an important question: If changing the loudness can have such an impact on the performance, to what extent is a good performance (according to a standard evaluation) simply caused by the "right" level of loudness?

B. Severe Impact on Evaluation

There are mainly three reasons why an evaluation as done in the previous section is less meaningful for algorithms that lack loudness-invariance.

First, the conclusion that a method is robust against false positives for data sets containing mostly specific instruments

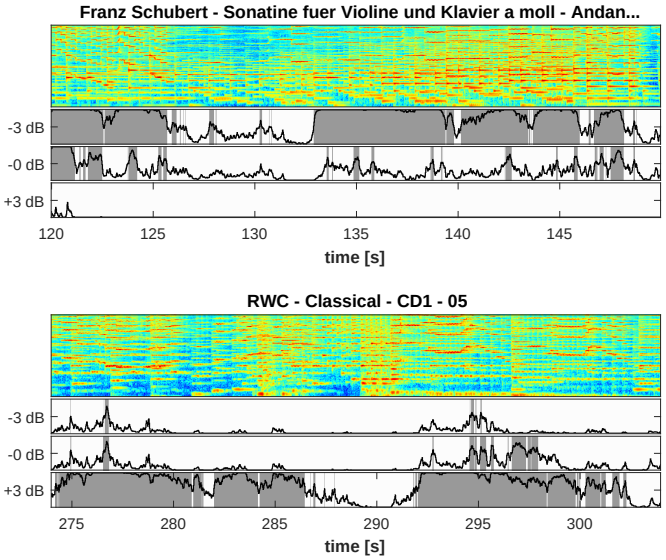

Fig. 8. Two examples of loudness sensitivity from the same model on pure instrumental music. Upper plots: log scaled spectrograms of each audio segment; below: posterior probabilities from decreased (-3 dB), untouched (-0 dB), and increased (+3 dB) loudness. Darker regions in the posterior probability plots indicate false positives.

is invalid. A good performance could just be the result of the most appropriate level of loudness.

Second, results may not accurately represent the behaviour that one could expect from a method on data taken "from the wild": recordings from social media platforms, where the recording conditions are different all the time, or even change throughout the same recording e.g. due to altering microphone positions.

Third, summarised results from a data set after all recordings were modified identically in terms of loudness may not reveal an existing weakness in this regard. As already shown (see Fig. 8), different levels of loudness can cause either lower or higher numbers of errors, depending on the recording. Therefore, an averaged result over several identically modified recordings could be similar to the results of the untouched recordings, since lower and higher numbers of errors per recording could cancel each other out.

The assessment that would reveal any loudness sensitivity best has to take into account that possibly just a single recording gives a different performance. Maybe there is just one example that could help to detect a specific weakness, and we consider such examples that induce erroneous behaviour most valuable for gaining insights. Therefore, we suggest a new evaluation strategy.

C. Evaluation Strategy for Loudness-Invariance

We suggest a different approach for evaluation between several levels of loudness in order to reveal a potential sensitivity to loudness.

First, we have to define the range of gain that is applied in order to end up with loudness-modified recordings. We suggest steps of 3 dB in both directions up to 9 dB, that is, we

end up with six new versions of the recordings.[7] After that, we compute the range of best and worst accuracy from all versions – inclusive the untouched ones – *per recording*. The distribution of the ranges in accuracy from a data set can then be summarised by a box plot, representing the sensitivity to loudness.

This way we prevent positive and negative effects from compensating each other, ending up in no overall impact.

Furthermore, even if just a single recording gives away a sensitivity to loudness, it will not get smoothed out, but appear as an outlier. In our case these are the most important examples to gain insight about model behaviour.

However, this evaluation should be considered only additional – although important – information. It does not represent the overall performance, hence the results from a standard evaluation still need to be taken into account in order to get meaningful insights. Our suggested evaluation approach could be considered a *measure of certainty* of the results from a standard evaluation: the lower the sensitivity to loudness, the more meaningful the standard evaluation.

In the next section, we will discuss the results from our evaluation strategy. We will demonstrate that methods that achieved the exact same results on the standard evaluation can behave quite differently.

D. Results

The box plots presented in Fig. 9 and 10 reflect the impact in accuracy between worst and best case, with a possible range of $0 - 100\%$, separately for validation and test sets. In order to interpret these, one has to consider two things: first, the lower the median and smaller the interquartile range of the distributions, the lower the sensitivity to loudness; second, outliers can be the most important examples in order to prove that a method is not loudness invariant. After all, we only need one example to disprove a hypothesis.

It can be seen that our new method is not affected by loudness manipulations, as designed, while all others are.

Furthermore, comparing CN1 and CN2, we can see that the data augmentation used to train CN2 is not only insufficient to eliminate loudness sensitivity, but even has a negative effect on some instrumental data sets (*hi, pakarina, sq, yt_heavy_instr, yt_wind_sax*). The obvious idea of additionally augmenting training data by random loudness gains within $\pm 10\,\mathrm{dB}$ [20, Sec. 4.2] does not change these results.

Interestingly, although (according to the standard evaluation) two methods yield the same results on the untouched audio recordings on the data set *softj* (LEH: 99.5%, NEW: 99.4%), our proposed evaluation reveals quite a difference between them in Fig. 9. It seems that the small number of false positives for method LEH is just the result of the right level of loudness, and the standard evaluation results can not be fully trusted. Similarly, in Fig. 10, the two methods give the same results on the untouched audio recordings on the data set *rwc_pop* (LEH: 87.7%, NEW: 87.7%), yet exhibit a different behaviour when evaluated on loudness modified audio recordings. Although the difference is not as clear as with the previous example, the median impact of loudness modification on accuracy is a lot higher (LEH: 7.4%, NEW: 0%). Another example is the data set *msd100*, where methods CN1 and NEW both reach 78.4% accuracy, and yet the loudness sensitivity is higher for CN1.

[7]Note that the gain should be applied to floating-point signals or features; for 16-bit integer samples a positive gain might lead to overflow or clipping and confound the results.

VI. Conclusion

This article has presented three contributions to the problem of singing voice detection from audio: an efficient and light-weight detection method that competes successfully with the state of the art; several new annotated data sets made publicly available; and a discussion and demonstration of a loudness dependence problem, along with a strategy for analysing it.

Only the feature learning approach [21, Sec. 9.8] with data augmentation (CN2) outperforms our method on songs, and reaches equal performance on instrumental music according to a standard evaluation. However, as demonstrated in a follow up experiment, CN2 exhibits a loudness sensitivity. This brought to light that part of the performance gap between CN2 and our approach is coincidentally due to a convenient level of loudness during standard evaluation.

We hope that this work will foster an understanding of the challenges and pitfalls related to developing and evaluating VD algorithms, and influence the way they are designed and evaluated in the future.

Acknowledgment

This research is supported by the Austrian Science Fund (FWF) under grants TRP307-N23 and Z159 (Wittgenstein Award), and by the Vienna Science and Technology Fund under grants NXT17-004, MA14-018. We also gratefully acknowledge the support of NVIDIA Corporation with the donation of two Tesla K40 GPUs and a Titan Xp GPU used for this research.

References

[1] G. W. Records, "Greatest vocal range, male," Website, 2016, available online at http://www.guinnessworldrecords.com/; visited on June 1st, 2017.

[2] L. Regnier and G. Peeters, "Singing voice detection in music tracks using direct voice vibrato detection," in *Proc. of the Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2009, pp. 1685–1688.

[3] M. Mauch, H. Fujihara, K. Yoshii, and M. Goto, "Timbre and Melody Features for the Recognition of Vocal Activity and Instrumental Solos in Polyphonic Music," in *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, 2011, pp. 233–238.

[4] A. L. Berenzweig and D. P. W. Ellis, "Locating singing voice segments within music signals," in *Workshop on the Applications of Signal Processing to Audio and Acoustics*. IEEE, 2001, pp. 119–122.

[5] Y. Li and D. Wang, "Separation of singing voice from music accompaniment for monaural recordings," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1475–1487, 2007.

[6] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[7] M. Ramona, G. Richard, and B. David, "Vocal detection in music with support vector machines," in *Proc. of the Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2008, pp. 1885–1888.

[8] A. De Cheveigné and H. Kawahara, "YIN, a fundamental frequency estimator for speech and music," *The Journal of the Acoustical Society of America*, vol. 111, pp. 1917–1930, 2002.

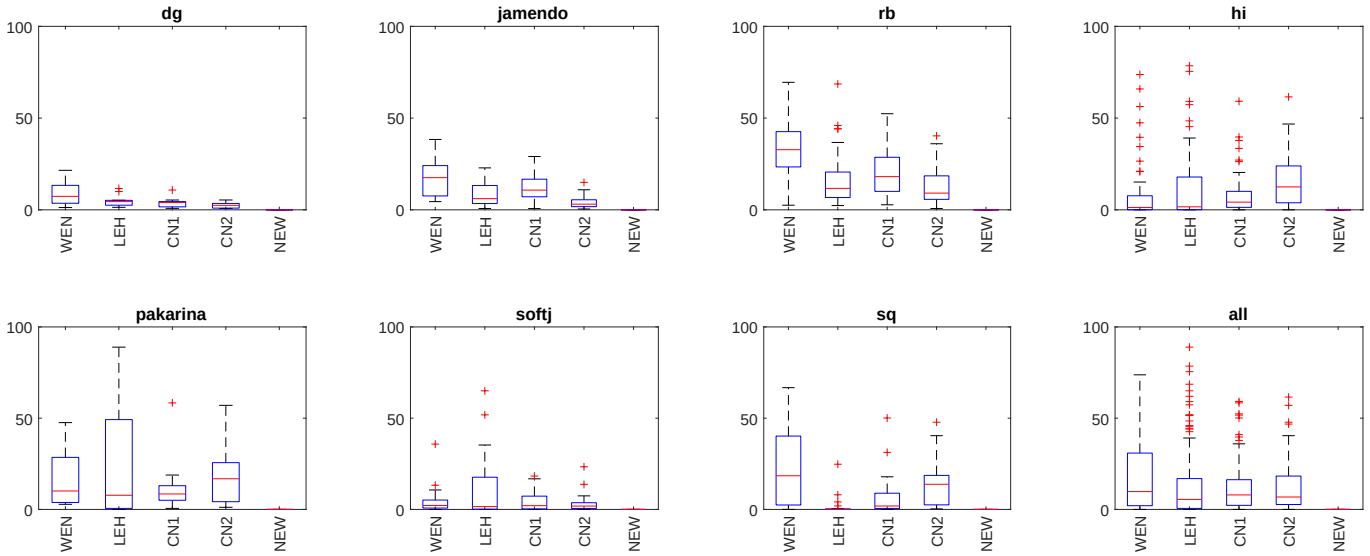

Fig. 9. The impact of different levels of loudness on the validation set (low values reflect loudness invariance). Notice that the instrumental data set *softj* gives equal performance for method LEH and NEW when conventionally evaluated, although they behave differently according to our suggested evaluation strategy. Our proposed method (NEW) is the only one not affected by varying levels of loudness.

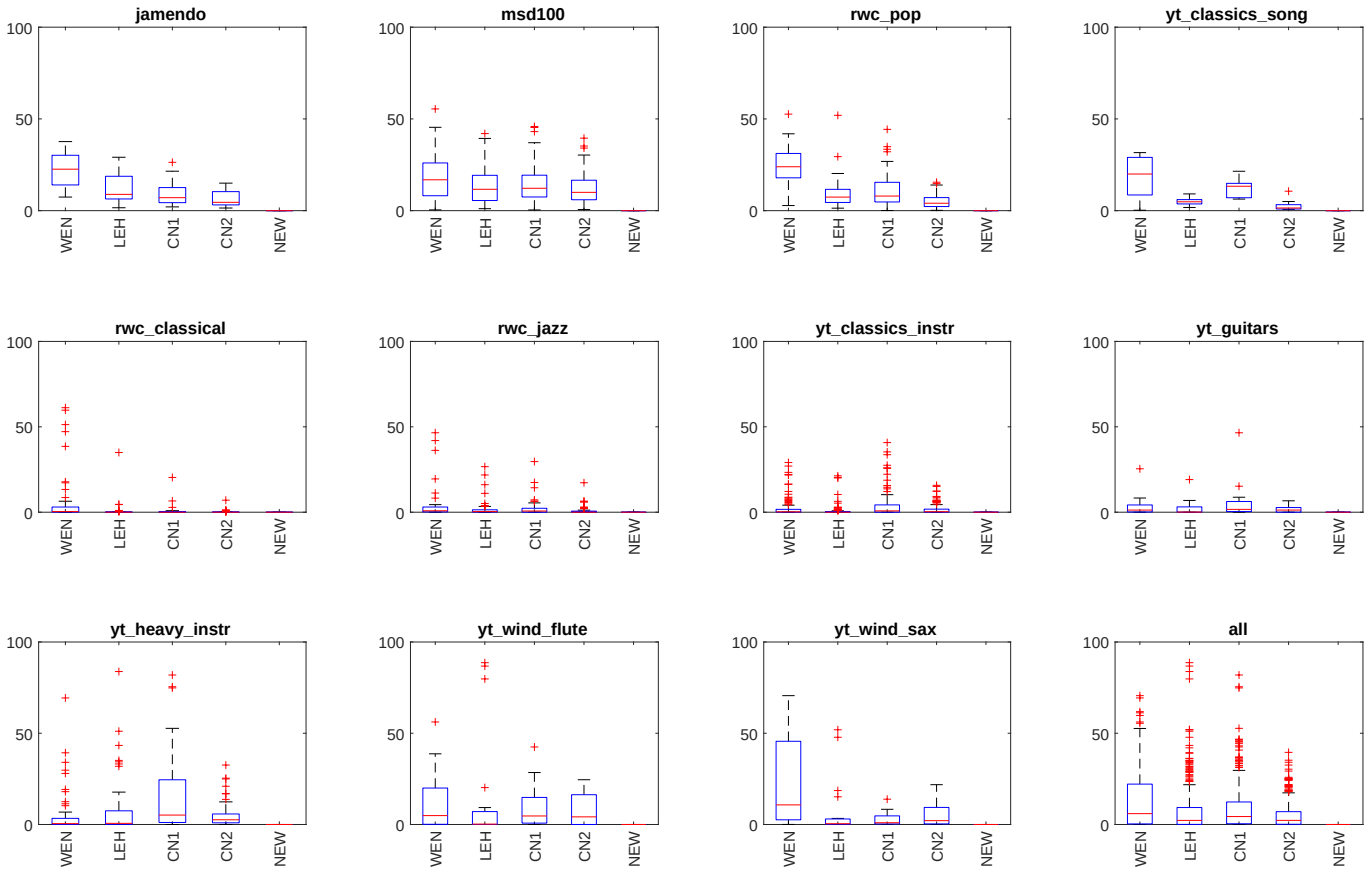

Fig. 10. The impact of different levels of loudness on the test set. Notice that the data set *rwc_pop* gives equal performance for method LEH and NEW when conventionally evaluated, although they behave differently according to our suggested evaluation strategy. Notice that the results of the middle row suggest similar sensitivity of all methods, although in general they behave quite differently. This supports our claim, that a thorough evaluation always has to be done with many data sets with different characteristics in terms of the challenges that they pose for VD algorithms.

[9] G. Peeters, "Automatic Classification of Large Musical Instrument Databases Using Hierarchical Classifiers with Inertia Ratio Maximization," in *115th AES Convention*, 2003. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=12460
[10] M. Goto, "A real-time music-scene-description system: Predominant-F0 estimation for detecting melody and bass lines in real-world audio signals," *Speech Communication*, vol. 43, no. 4, pp. 311–329, 2004.
[11] Y. Altun, I. Tsochantaridis, T. Hofmann *et al.*, "Hidden Markov support vector machines," in *Proc. of the Int. Conf. on Machine Learning (ICML)*, vol. 20, 2003, pp. 3–10.
[12] T. Joachims, T. Finley, and C. J. Yu, "Cutting-plane training of structural SVMs," *Machine Learning*, vol. 77, no. 1, pp. 27–59, 2009.
[13] F. Weninger, M. Wöllmer, and B. Schuller, "Automatic assessment of singer traits in popular music: Gender, age, height and race," in *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, 2011.
[14] F. Eyben, M. Wöllmer, and B. Schuller, "Opensmile: the munich versatile and fast open-source audio feature extractor," in *Proc. of the Int. Conf. on Multimedia*. ACM, 2010, pp. 1459–1462.
[15] C.-L. Hsu, D. Wang, J.-S. R. Jang, and K. Hu, "A Tandem Algorithm for Singing Pitch Extraction and Voice Separation From Music Accompaniment," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 20, no. 5, pp. 1482–1491, 2012.
[16] B. Lehner, R. Sonnleitner, and G. Widmer, "Towards Light-weight, Real-time-capable Singing Voice Detection," in *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, 2013, pp. 53–58.
[17] B. Lehner, G. Widmer, and S. Böck, "A low-latency, real-time-capable singing voice detection method with lstm recurrent neural networks," in *Proc. of the European Signal Processing Conf. (EUSIPCO)*. IEEE, 2015, pp. 21–25.
[18] B. Lehner, G. Widmer, and R. Sonnleitner, "On the reduction of false positives in singing voice detection," in *Proc. of the Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2014, pp. 7530–7534.
[19] S. Leglaive, R. Hennequin, and R. Badeau, "Singing voice detection with deep recurrent neural networks," in *Proc. of the Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2015, pp. 121–125.
[20] J. Schlüter and T. Grill, "Exploring data augmentation for improved singing voice detection with neural networks," in *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, 2015, pp. 121–126.
[21] J. Schlüter, "Deep Learning for Event Detection, Sequence Labelling and Similarity Estimation in Music Signals," Ph.D. dissertation, Johannes Kepler University Linz, Austria, Jul. 2017.
[22] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[23] A. Graves and J. Schmidhuber, "Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks," in *Advances in Neural Information Processing Systems (NIPS)*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds. Curran Associates, Inc., 2009, pp. 545–552.
[24] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. of the Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2013, pp. 6645–6649.
[25] B. Bogert, M. Healy, and J. Tukey, "The quefrency alanysis of time series for echoes: cepstrum, pseudo-autocovariance, cross-cepstrum, and saphe cracking," in *Proceedings of the Symposium on Time Series Analysis*, M. Rosenblatt, Ed. New York: John Wiley and Sons, 1963, pp. 209–243.
[26] C. Dittmar, B. Lehner, T. Prätzlich, M. Müller, and G. Widmer, "Cross-version singing voice detection in classical opera recordings." in *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, 2015, pp. 618–624.
[27] B. Lehner and G. Widmer, "Monaural blind source separation in the context of vocal detection." in *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, 2015, pp. 309–315.
[28] R. Sonnleitner, B. Niedermayer, G. Widmer, and J. Schlüter, "A Simple And Effective Spectral Feature For Speech Detection In Mixed Audio Signals," in *Proc. of the Int. Conf. on Digital Audio Effects (DAFx)*, 2012.
[29] W. A. Sethares, *Rhythm and Transforms*. Springer, 2007.
[30] J. Gray, A. and J. Markel, "A spectral-flatness measure for studying the autocorrelation method of linear prediction of speech analysis," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 22, no. 3, pp. 207–217, Jun 1974.
[31] C. M. Bishop, *Pattern Recognition and Machine Learning*, 11th ed., ser. Information Science and Statistics. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.
[32] P. Gérard, L. Kratz, and S. Zimmer, "Jamendo, open your ears," Website, 2005, available online at http://www.jamendo.com; visited on August 1st, 2017.
[33] A. Flexer and D. Schnitzer, "Effects of album and artist filters in audio similarity computed for very large music databases," *Computer Music Journal*, vol. 34, no. 3, pp. 20–28, 2010.
[34] Y. E. Kim, D. S. Williamson, and S. Pilli, "Towards Quantifying the "Album Effect" in Artist Identification." in *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, 2006, pp. 393–394.
[35] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC music database: Popular, classical, and jazz music databases," in *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, vol. 2, 2002, pp. 287–288.
[36] N. Ono, Z. Rafii, D. Kitamura, N. Ito, and A. Liutkus, "The 2015 signal separation evaluation campaign," in *Int. Conf. on Latent Variable Analysis and Signal Separation*. Springer, 2015, pp. 387–395.
[37] F. Weninger, J.-L. Durrieu, F. Eyben, G. Richard, and B. Schuller, "Combining monaural source separation with long short-term memory for increased robustness in vocalist gender recognition," in *Proc. of the Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2011, pp. 2196–2199.
[38] A. Graves, "RNNLIB: A recurrent neural network library for sequence learning problems," Website, 2013, available online at https://sourceforge.net/projects/rnnl/; visited on June 1st, 2017.

Bernhard Lehner received the B.S. and M.S. degrees in computer science from Johannes Kepler University, Linz, Austria, in 2007 and 2010, respectively, where he is currently pursuing the Ph.D. degree. He was with VA Tech, Lenze, Siemens, and Infineon, from 1991 to 2004. His research interests include signal processing, audio event detection, audio scene classification, music information retrieval, image processing, neural networks, and interpretable machine learning.

Jan Schlüter has been pursuing research on deep learning for audio processing since 2010, currently as a postdoctoral researcher at the Austrian Research Institute for Artificial Intelligence (OFAI), Vienna, Austria. He earned his PhD at the Johannes Kepler University, Linz, Austria in 2017. His research interests include machine listening, signal processing and deep learning. He also co-authors and maintains the open source deep learning library "Lasagne".

Gerhard Widmer is Professor and Head of the Department of Computational Perception at Johannes Kepler University, Linz, Austria, and leads the Intelligent Music Processing and Machine Learning Group at the Austrian Research Institute for Artificial Intelligence (OFAI), Vienna, Austria. His research interests include AI, machine learning, and intelligent music processing, and his work is published in a wide range of scientific fields, from AI and machine learning to audio, multimedia, musicology, and music psychology. He is a Fellow of the European Association for Artificial Intelligence (EurAI), and has been awarded Austrias highest research awards, the START Prize (1998) and the Wittgenstein Award (2009). He currently holds an ERC Advanced Grant for research on computational models of expressivity in music.