applied sciences

MDPI

Article

Exploring Channel Properties to Improve Singing Voice Detection with Convolutional Neural Networks

Wenming Gui [1,2,*], Yukun Li [3], Xian Zang [1] and Jinglan Zhang [4]

[1] School of Software Engineering, Jinling Institute of Technology, Nanjing 211169, China; zangxian_friend@126.com
[2] Key Lab of Broadband Wireless Communication and Sensor Network Technology (Nanjing University of Posts and Telecommunications), Ministry of Education, Nanjing 210003, China
[3] Centre for Digital Music, Queen Mary University of London, London E1 4NS, UK; yukun.li@qmul.ac.uk
[4] Faculty of Science, Queensland University of Technology, Brisbane, QLD 4001, Australia; jinglan.zhang@qut.edu.au
* Correspondence: guiwenming@126.com

Abstract: Singing voice detection is still a challenging task because the voice can be obscured by instruments having the same frequency band, and even the same timbre, produced by mimicking the mechanism of human singing. Because of the poor adaptability and complexity of feature engineering, there is a recent trend towards feature learning in which deep neural networks play the roles of feature extraction and classification. In this paper, we present two methods to explore the channel properties in the convolution neural network to improve the performance of singing voice detection by feature learning. First, channel attention learning is presented to measure the importance of a feature, in which two attention mechanisms are exploited, i.e., the scaled dot-product and squeeze-and-excitation. This method focuses on learning the importance of the feature map so that the neurons can place more attention on the more important feature maps. Second, the multi-scale representations are fed to the input channels, aiming at adding more information in terms of scale. Generally, different songs need different scales of a spectrogram to be represented, and multi-scale representations ensure the network can choose the best one for the task. In the experimental stage, we proved the effectiveness of the two methods based on three public datasets, with the accuracy performance increasing by up to 2.13 percent compared to its already high initial level.

Keywords: singing voice detection; convolutional neural network; scaled dot-product; squeeze-and-excitation; multi-scale channels

Citation: Gui, W.; Li, Y.; Zang, X.; Zhang, J. Exploring Channel Properties to Improve Singing Voice Detection with Convolutional Neural Networks. *Appl. Sci.* **2021**, *11*, 11838. https://doi.org/10.3390/app112411838

Academic Editor: Philippe Esling

Received: 20 October 2021
Accepted: 9 December 2021
Published: 13 December 2021

1. Introduction

Singing voice detection (SVD) is a task used to discriminate whether an audio segment contains at least one person's singing voice. Figure 1 illustrates the annotation of singing voices of a music clip through SVD, where the yellow segments indicate singing voices. Although it is very easy for humans to identify the vocal component in songs, this task is still difficult for machines. Clearly, there are a large number of variations of singing voices and instruments in musical signals. In particular, some instruments mimic the mechanism of human singing, and share the same frequency bands and the timbre characteristics as those of the singing voice. This increases the difficulty of the SVD task. In the field of music information retrieval (MIR), SVD is fundamental and crucial for various tasks, such as singing voice separation [1], singer identification [2], and lyrics alignment [3].

In general, the SVD task is addressed via two key steps: feature extraction and classification. The feature extraction step conducts signal transformation, dimensionality reduction, etc., to produce features by which it could be easy to identify the singing voice from a song; for example, Linear Prediction Coefficients (LPCs) [4] and Mel-Frequency Cepstral Coefficients (MFCCs) [5]. The classification step categorizes music segments into

voice and non-voice classes using a classifier with the extracted feature as the input. Several classifiers have been employed, from the traditional methods, such as Hidden Markov Models (HMM) [4,6], Gaussian Mixture Models (GMM) [5], Support Vector Machines [7], and Random Forests [8], to the modern Deep Neural Networks (DNN), such as Convolution Neural Networks (CNNs) [9–13], and Recurrent Neural Networks (RNNs) [14,15]. In addition to the feature extraction and classification steps, the preprocessing and postprocessing steps are also beneficial to the results. Singing voice separation is treated as the main preprocessing method for the SVD task [15,16]. Regarding postprocessing, Zhang et al. [16] addressed two methods: the median filter and HMM.

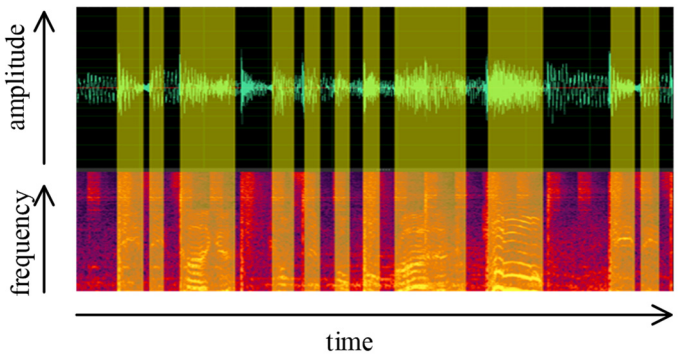


Figure 1. An illustration of singing voice detection. The upper image is the waveform of a music clip, and the lower image is the spectrogram.

To date, the majority of SVD algorithms have focused on feature engineering, and early features drew on the experience of speech recognition. In addition to LPCs and MFCCs mentioned above, many other speech recognition-inspired features have been proposed for SVD, such as Perceptual Linear Prediction Coefficients (PLPCs) [17], Zero-Crossing Rate (ZCR), Spectral Flux (SF), Harmonic Coefficients (HCs), and Log Frequency Power Coefficients (LFPCs) [6]. Rocamora et al. compared the six features (MFCCs, LFPCs, SF, PLPCs, HCs, and Pitch) based on their delta and double delta coefficients, and found that MFCCs were the most effective features for SVD. However, in the literature, the singing voice is usually represented by the combination of a number of features, rather than a single feature aiming at describing its characteristic from the different aspects. For example, a combination of four features, namely, MFCCs, LPCs, PLPs, and HCs, were employed by Li and Wang [18] to detect the singing voice, where an HMM was used as a classifier. Ramona et al. utilized a more comprehensive combination of the features including MFCCs, LPCs, ZCR, sharpness, spread, f0, and other descriptors, and then fed these into the SVM classifier [19].

Early features identified singing voice as speech, treating the SVD task as discriminating the human voice from a natural environment, in which researchers did not highlight the musical characteristics. Recently, however, researchers have begun to explore the musical characteristics of the singing voice, developing features and adding them as the parts of the feature combination. Regnier et al. presented a SVD method using the vibrato and tremolo parameters, generated from the selection of partials extracted from the audio [20]. It was supposed that the singing voice was characterized by harmonicity, formants, vibrato, and tremolo. This idea originated purely from the musical characteristics. Mauch et al. combined four features [21] for vocal activity detection. These features were not only drawn from speech recognition, but also from pitch fluctuation and melody. Similarly, Lehner et al. added the Fluctogram, which is a discrete measure of pitch fluctuation, and reliability indicators, to the feature combination [11], and argued that the algorithm was loudness invariant. Zhang et al. also employed the feature from the music characteristic, i.e., Chroma, in his bucket, which was composed of eight different features [16].

As a result of feature engineering plus powerful classification capability, DNNs have resulted in state-of-the-art SVD performance. Lehner proposed a state-of-the-art method, based on a LSTM (Long Short-Term Memory) RNN, having 29 features including the Fluctogram and MFCCs. Schlüter presented state-of-the-art performance based on a CNN with data augmentation [10]. CNNs can effectively work in the spatial domain due to the convolution between neurons and feature maps, which can enable good performance in the image processing field. However, because the CNNs are not capable of memorization, they are not suitable for processing sequences associated with time. In [16], Zhang et al. tried to employ a LRCN (Long-Term Recurrent Convolutional Network) to integrate the advantages of CNN and RNN, i.e., spatially learning features via CNN and temporally learning the relationships via RNN. In their algorithm, they fed the feature engineered vector, and the 288 combined coefficients, to the LRCN. With the help of the singing voice separation preprocessing step and the HMM-based postprocessing step, they obtained very good results on the public datasets.

DNNs are not only capable of classification, but also feature learning. Therefore, the feature learning capability of DNNs can be used to undertake the work of feature engineering in the SVD framework, and, ultimately, this step can even be eliminated. Feature engineering has a large number of advantages, including interpretability, deterministic nature, and low computation cost. However, every coin has two sides. One of the disadvantages of feature engineering is that the artificial feature has poor adaptability. Thus, a new dataset or situation requires new engineering, such as in the case of designing reliability indicators for the Fluctogram [11]. Another disadvantage is that, when elaborating the complicated high-level features for the singing voice characteristics, some useful information is lost because high-level features are irreversible to the original signal. As a result, the DNN cannot be trained to find that useful information. Therefore, there is a trend towards feature learning with DNNs in which only low-level representation is fed to them. In this way, Schlüter [10,22] used a CNN and Leglaive used an RNN [15]. The input of the former was the mel spectrogram (in the current study, we identified the macroscopic spectrogram with the microscopic image from the spectrogram fed to the DNN), whereas that of the latter was the log-mel spectrogram. Only slight differences were seen in the two approaches, which achieved performances on par with the state-of-the-art methods using feature engineering.

In this study, we focused on exploring channel properties to improve the SVD performance based on the condition that a CNN is used for feature learning with low-level representation input, i.e., the log-mel spectrogram. CNNs work spatially and the channels hold the core information residing on the layers of the CNNs. As specific channels, feature maps represent the features learned by the convolution neurons. It is worth noting that the channel properties in CNNs consist not only of properties of channels on the inner layers, but also those in the input channels. We proposed two methods to improve the CNN performance in terms of the channel. First, channel attention is presented to measure the importance of the feature, and two attention mechanisms are exploited to learn the weights of importance. Second, multi-scale representations are fed to the input channels, with the aim of adding more information in terms of scale to the CNN.

The remainder of the paper is organized as follows. Our methods, including channel attention and multi-scale input channels, are introduced in Section 2. In Section 3, the improvements related to CNN architecture are described. We show our experiments and results in Section 4. Finally, the conclusion and discussions are presented in Section 5.

2. Methods

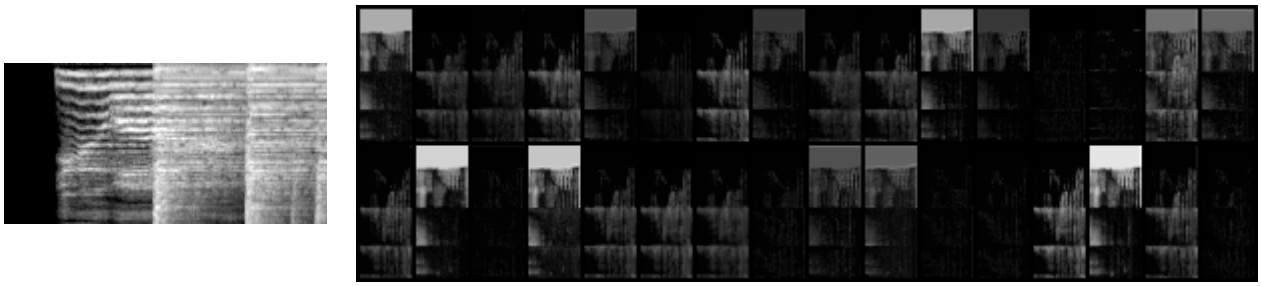
In this section, we discuss the proposed methods in terms of CNN channels: channel attention and multi-scale channels. For each method, we first describe the motivation and then describe the method in detail. We also present related work concerning the method. We show the experimental results related to the two methods in Section 4.

2.1. Channel Attention

We first introduce the motivation for channel attention with an example, and discuss the related work. Then we present the two channel attention mechanisms for the SVD task: the scaled dot-product and squeeze-and-excitation.

2.1.1. Motivation

Because we concentrate on learning features rather than designing features, it is necessary to study how to improve the learning capability. In the CNN, feature maps on the inner layers hold the core information about the original signal. In the conventional CNN, the convolutional neurons equally "observe" the feature maps on the upper layer, in which "equally" means all the feature maps on the upper layer assign the same weights to the neurons. This implies that those features would play equally important roles in achieving the classification results. We show an example in Figure 2. From the figure, we can see that the black dominated feature maps, for example, the 2nd, 3rd, 4th, 6th, 7th, 9th, and 10th feature maps on the first row, contain little information for detecting the singing voice; they thus need to be restrained or neglected. Therefore, we can draw the conclusion that some feature maps need to be restrained or deleted so that the neurons can focus their "attention" on the important feature maps.



(**a**) (**b**)

Figure 2. The conventional CNN neurons equally "observe" the feature maps on the upper layer. (**a**) An image from the log-mel spectrogram of the music "you are", and (**b**) a group of the feature map examples from a CNN inner layer with 64 channels.

To measure the "attention" of a feature map that the neurons should impose, we can design a mechanism in which the attention is adjusted by learning. Afterwards, the convolutional neurons can distinctively "observe" the feature maps in terms of importance. In Figure 3, we illustrate the learning mechanism for CNNs. It is supposed that there are two adjacent layers—the layer n1 and the layer n—in a CNN; the layer $n-1$ is above the input, and $F = (F_1, F_2, \ldots F_m)$ are the output feature maps at the layer $n-1$, where every channel F_i presents the corresponding feature map, and is actually an image with the same height h and width w. In this architecture, we try to learn the weight w_i as the attention of the channel F_i, and then feed the weighted channels $F' = FW$ to the next layer n, where $W = (w_1, w_2, \ldots w_m)^T$. After imposing the attention values, the feature maps at layer $n-1$ can be treated more distinctively for the results. It was hoped that this mechanism can improve the overall SVD performance.

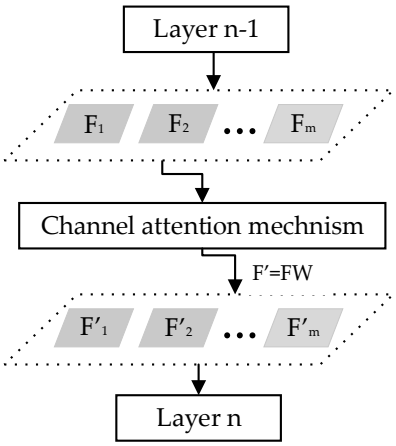


Figure 3. Channel attention mechanism.

2.1.2. Related Work

We organize the related work in terms of three aspects: the origin of attention mechanism, the attention mechanism in the RNNs, and the attention mechanism in the CNNs.

1. The origin of attention mechanism

To the best of our knowledge, the first work in which the attention mechanism was associated with the Neural Network (NN) was undertaken by Graves. In [23], he proposed the Neural Turing Machine (NTM), in which a content-based addressing mechanism was employed to impose "blurry" read and write operations on the "working memory". Taking the read operation as an example, the operation at the time t can be simply formulated as the following equation:

$$R_t = softmax(g(k, M))M \tag{1}$$

where $M = (m_1, m_2, \ldots\ m_n)$ is the row vector of the memory contents, k is the key vector produced by the read head, and g is a cosine function to measure the similarity between the key vector u and the content v:

$$g(u, v) = \frac{u \cdot v}{||u|| \cdot ||v||} \tag{2}$$

Other than the traditional read operation, in which the address and the returned value are fixed at time t, the address is blurry, depending on the distribution on all the memories, and the returned value is contingent on all the memory contents and their corresponding weights. This addressing mechanism essentially is an attentional process in which the weight refers to the degree of attention and the cosine similarity is used to measure the degree. Many attention mechanisms have emerged since Graves' NTM was proposed, especially in the Natural Language Processing (NLP) field with RNNs.

2. The attention mechanism in the RNNs

In the NLP field, Bahdanau et al. [24] proposed an additive attention model to score how well the input words around position j and the output at the position i matched, and then the annotations were computed by these scores. Finally, the context vector for each target word was obtained by the weighed sum of these annotations. Luong et al. [25] further presented a series of attention mechanisms to address the problem, categorized as the global attention and the local attention. The former placed the attention on all the source words, as in the approach of Bahdanau et al., but with some different perspectives, whereas the latter solely focused on the small window of inputs which were differentiable. Vaswani et al. [26] proposed the other well-known attention mechanism, named self-attention. As the word "self" implies, the attention created a mutual relationship among the source words, which was different from the relationship of the other mechanisms, and

was described as being between the target words and the source words. In this mechanism, the scaled dot-product (SDP) was utilized to measure the degree of the relationship:

$$\text{Attention}(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{3}$$

where Q, K, and V represent the matrix form of query, key, and value, respectively, and d_k is the key dimension. Similar to the content-based addressing mechanism [23], SDP first calculates the similarity measure and then transforms this measure to the weight matrix by the *softmax* function. Finally, V is weighted by the attention. Moreover, to learn more relative information, they offered a multi-head model to represent the different subspaces:

$$\begin{aligned} \text{Multihead}(Q, K, V) &= \text{Concat}(\text{head}_1, \cdots, \text{head}_h)W^O \\ \text{where head}_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \end{aligned} \tag{4}$$

3. The attention mechanism in the CNNs

To date, most of the attention mechanisms have been designed for RNNs. As another an important variant of DNNs, the CNN plays a signficant role in audio and video processing. Can we place attention on the CNN and improve its performance? Hu et al. [27] proposed a squeeze-and-excitation block to recalibrate the relations among the channels, and achieved great success in a recent image classification task. Other researchers [28,29] have tried to integrate channel and spatial attention to solve the image processing problems. Unlike RNNs, CNNs have a hierarchy of distinct feature maps, and there are different feature maps at different layers which are generated through different convolutional kernels.

2.1.3. Channel Attention Based on a Scaled Dot-Product Module

The scaled dot-product in the self-attention mechanism [26] is usually utilized in RNNs for NLP. To the best of our knowledge, no research has been undertaken using this mechanism in CNNs for SVD. Here, we amended our channel attention mechanism and integrated it into a CNN to learn the importance of the features. Figure 4 shows the module framework. F represents the feature maps with the c channels of images input to the module, whose height is h and width is w. F′ are the module outputs with the same size of channels. Four steps are involved in learning the channel attention.

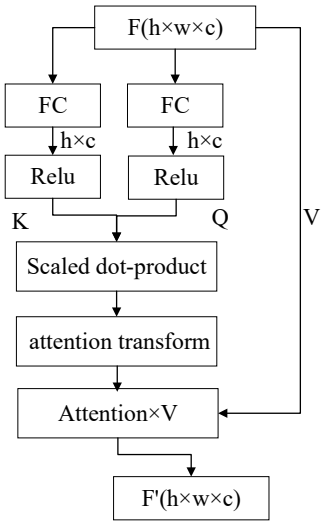


Figure 4. Channel attention based on a scaled dot-product module.

First, we linearly project the input feature maps with dimensions reduced to h, and then feed them to the ReLU functions to obtain keys and queries:

$$K = \sigma(FW^K) \qquad Q = \sigma(FW^Q) \tag{5}$$

where the projections are parameter vectors $W^K, W^Q \in R^w$, and $K, Q \in R^{h \times c}$. This step increases the nonlinear characteristic of ReLU functions, and the following computation complexity is reduced by reducing the dimensions.

Then, the attentions or the weights are computed by the scaled dot-product, followed by a *softmax* function:

$$A = softmax(\frac{Q^T K}{\sqrt{h}}) \tag{6}$$

where $A \in R^{c \times c}$ represents the correlation matrix of the channels, and h is the height of image.

Next, an attention transform is conducted to obtain the attention vector:

$$\alpha = \text{mean}(A(1 - E), \text{dim} = 1) \tag{7}$$

where $\alpha \in R^c$ and E is an identity matrix. $A(1 - E)$ aims at eliminating the relation between the channel and itself. The mean operation transforms the matrix into an attention vector.

Finally, the output feature maps are achieved by imposing attention on the value V:

$$F' = \alpha V \tag{8}$$

where V is equal to F. The output feature maps F' can be treated as the features embedding the importance, and they can then be fed into the next layer.

2.1.4. Channel Attention Based on a Squeeze-and-Excitation Module

The squeeze-and-excitation (SE) module was initially introduced by Hu et al. in the image classification field [27]. In addition to its use in studying the spatial structure of the CNN, the SE module strengthens its learning capabilities by making use of the dependencies among channels. This dependency is similar to our attention for the CNN; therefore, it was adopted to compute the channel attention or feature map importance. Our framework is shown in Figure 5.

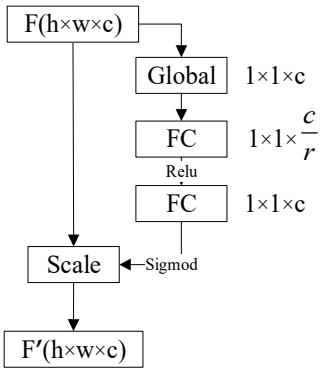


Figure 5. Channel attention based on a squeeze-and-excitation module.

At the first step, we squeeze each channel, i.e., feature map, into a channel descriptor using a global mean pooling function:

$$z = \text{mean}(F, \text{dim} = 2) \tag{9}$$

where dim = 2 means operation on the channel dimension and each channel size is h × w, where h is the height and w is the width. The vector $z \in R^c$ denotes the compact information of the original feature maps.

At the second step, a gating mechanism is carried out on the descriptor z aiming at limitation and generalization of the model, which comprises three units: a fully connected layer for reducing dimensionality, a ReLU, and a fully connected layer for increasing dimension. The sigmoid function is employed to extract the channel-wise weights, i.e., attention.

$$\alpha = \sigma(\delta(z^T W^z) W^\alpha) \tag{10}$$

where σ, δ are the sigmoid function and the ReLU function, respectively, $W^z \in R^{c \times r}$ and $W^a \in R^{r \times c}$, and c is a multiple of r.

Finally, the original feature maps, F, are rescaled by the attentions:

$$F' = \alpha F \tag{11}$$

In this way, the feature importance is achieved by the channel attention based on a squeeze-and-excitation operation. The feature maps are adjusted in terms of importance before they enter into the next layer of the CNN.

2.2. Multi-Scale Input Channels

We first explain our motivation for the multi-scale input channels method, and then introduce the related work. Finally, we present the method in detail.

2.2.1. Motivation

In this study, the aim was to improve the SVD performance using the CNNs with feature learning by feeding spectrograms. Here, there is a precondition that the spectrogram should be as correct and complete as possible to represent the original musical signal. The representation quality of the spectrogram depends mainly on the time resolution and the frequency resolution at the stage of STFT. It is worth noting that the time resolution and the frequency resolution restrict each other, and that the best frequency resolution cannot be obtained simultaneously with the best time resolution. In general, these two resolutions are contingent on the size of the window function of STFT. We refer to size or duration as "scale" in this paper. In practice, different musical signals need different scales because they have their own predominant frequency bands and time structures. The better the scale adapts to the signal, the better the representation. We show the typical scales, and the corresponding time and frequency resolutions, in the Table 1 below, where the signal sampling rate is 22,050 Hz.

Table 1. The scales and the corresponding time and frequency resolutions.

Scale	512	1024	1536	2048	2560	3072	3584	4096
Δt (ms)	23.2	46.4	69.7	92.9	116.1	139.3	162.5	185.8
Δf (Hz)	43.1	21.5	14.4	10.8	8.6	7.2	6.2	5.4

The frequency resolutions in the table above may be suitable for most songs. However, which is the best? The answer is different for different songs. The conventional method used to resolve this problem is to choose the best scale after testing all the scales on the datasets. We propose to combine the multi-scale spectrograms as the input channels. Given the multi-scale input channels, in which some representations may be suitable for the signal while others may be unsuitable, the CNNs can learn and choose the optimal representation for the task.

2.2.2. Related Work

Almost all of the work in the field of SVD chooses a pre-defined scale for the task, whether in CNNs or RNNs. In [11], the scale of the input channel of the LSTM was fixed with the length of 100 ms, and the length in the CNN was 46.4 ms [10].

In [16], although the scale was still fixed for the input channel of the LRCN, the comparison was conducted among the different scales, where the scales ranged from 50 ms to 2 s. It was found that the scale of 0.8 s was optimal on the Jamendo dataset, and that of 1 s was optimal on the RWC dataset. It was proved that the choice of scale had an important effect on the performance. However, we should note that this optimal scale may not be suitable for our method because, in their study, the singing voice separation was performed to preprocess the signal and the STFT was computed on the preprocessed vocal signal.

2.2.3. Method

In addition to one fixed scale spectrogram, we feed the combined channels with the multi-scale spectrograms to the CNN. It is up to the CNN to learn from the different scales of the input channels. Once the optimal representation of the original signal is found, the performance can be improved. We illustrate the different input channel scheme in Figure 6.

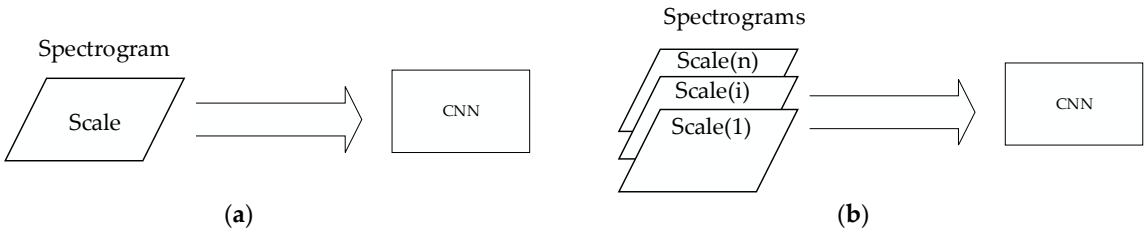


Figure 6. Single scale spectrogram input (**a**) and the multi-scale spectrograms' input (**b**).

When computing the different scale of spectrograms, we pad zeros on the left of the signal, where the number of zeros is determined according to the scale. The hop size is held constant so that the times of different scales are aligned. Therefore, the annotations of the different scales of the spectrograms are the same, ensuring that we do not need to change the other settings of the CNN. We show the example scales of 512, 2048, and 4096 in Figure 7. From the figure, the image of the smaller scale of 512 is a slightly blurry compared to the larger scales due to poor frequency resolution. However, because of poor time resolution, the white texture shown in the rectangle of the 4096 scale projects the line near 300 in Figure 7c, although this scale has the best frequency resolution among the three scales. Hence, the multi-scale spectrograms need to be fed to the CNN channels, and the CNN decides which spectrogram it should choose.

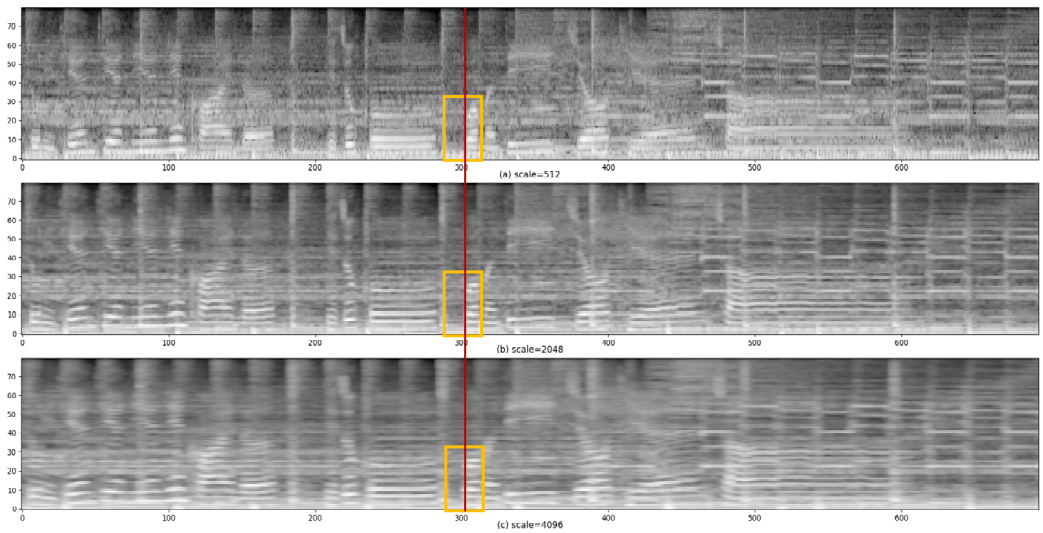


Figure 7. The log-mel spectrograms of the song mir1k_jmzen_2_12 on the scales of 512, 2048, and 4096.

3. Proposed CNN Architecture

In this section, we first describe the baseline CNN architecture characterized by feature learning, rather than feature engineering, and then we describe the architecture improvements. Finally, we present our proposed architecture.

3.1. Baseline Architecture

In this study, we used the CNN architecture from [22] as the baseline, where the CNN consists of four CNN layers and three dense layers, as shown in Figure 8. Each CNN layer can extract the features by "observing" the upper layer feature maps. Theoretically, the deeper the stack of the CNN network, the more accurate the extracted features. This network is relatively shallower because the experimental datasets are relatively smaller. The 3×3 convolutional kernel is applied to each of the four CNN layers, followed by a Leaky ReLU activation function. The receptive field of 3×3 is relatively smaller, but the stack of two 3×3 layers has an effective receptive field of 5×5. The stacked smaller receptive field layers can make the decision function more discriminative and decrease the number of parameters [30]. Because rectifier nonlinearities can improve neural network acoustic models, Leaky ReLU is slightly better than ReLU [31]. Therefore, Leaky ReLU was employed in the architecture. The two non-overlapping 3×3 max pooling stages follow the two CNN stacks, respectively, to reduce the image size and to avoid over-fitting to some extent. The two dropouts also play the role of mitigating over-fitting, whereas the dense layers transform feature maps into feature vectors for classification.

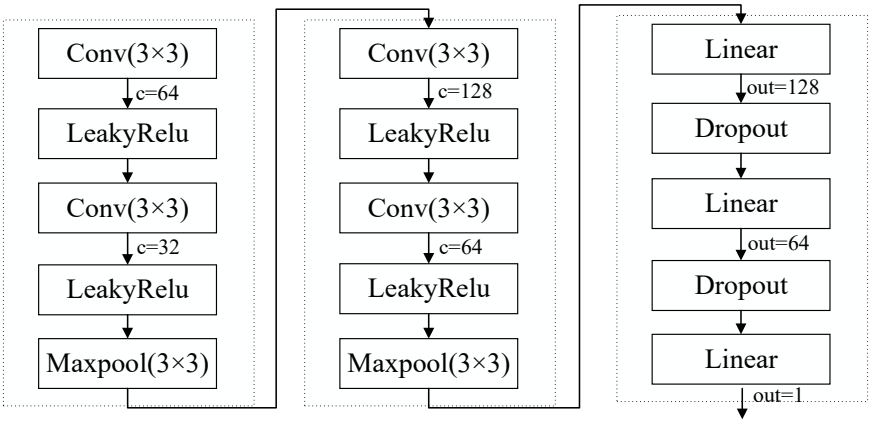


Figure 8. The baseline CNN architecture.

In this architecture, the CNN is only fed with a simple log-mel spectrogram, without feature engineering, data augmentation, preprocessing, or postprocessing. This is desirable because the CNN, rather than feature engineering, is required to learn the features.

3.2. The Improvements

Three improvements were made to the baseline architecture.

First, the spectrograms were normalized by the precomputed min and max matrices before they entered the CNN. The min and max matrices were calculated in terms of image intensity, based on all the images generated from the spectrograms of the training, validation, and test datasets. Conventionally, we obtain the min and max scalars for normalization instead of their matrix counterparts; for example, for all the images $m_k = [v^k_{i,j}], k \in [1, K], i \in [1, I], j \in [1, J]$, where $v_{i,j}$ is the intensity at the position (i, j). Conventionally, we also obtain the min scalar value $p = \min_{k\in[1,K],i\in[1,I],k\in[1,J]}(v^k_{i,j})$ for min–max normalization; however, in this study we obtain the min matrix $\mathrm{P} = [p_{i,j}] = [\min_{k\in[1,K]}(v^k_{i,j})]$,

$i \in [1, I], j \in [1, J]$. This technique eliminated the effects of the variation in the scale in terms of intensity. We considered this to be more precise in terms of the overall spectrogram.

Second, a Batch Normalization layer was added after every convolution layer. The Batch Normalization layer can reduce the internal covariate shift of the network and increase the speed of training [32].

Third, we employed weighted binary cross entropy as the loss function. This is weighted because the overall duration of the singing voice is usually longer than that of non-singing in popular songs, and the application of SVD usually concerns popular music. We set the weight vector as the proportion of the number of voiced and unvoiced samples in the training, validation, and test datasets. The loss function can be formulated as follows:

$$l = -\frac{1}{N}\sum_{i} w_i[y_i \log x_i + (1 - y_i)\log(1 - x_i)] \tag{12}$$

where $i \in [1, N]$, N is the number of samples, x_i is the probability of voice, and y_i is the sample label.

3.3. Proposed CNN Architecture

The overall proposed CNN architecture is shown in Figure 9, in which CAM refers to channel attention module. The CAM module is either the scaled dot-product module or the squeeze-and-excitation module. This means the two channel attention steps are integrated with the original CNN to measure the importance of the upper feature maps. As an implementation of the multi-scale input method, we feed the n scales of spectrograms at the same time to the input channels, instead of only one scale of a spectrogram. Furthermore, to concisely illustrate the architecture, we group a convolution, a BatchNorm, and a Leaky ReLU as the BNConv block, as shown in Figure 9a.

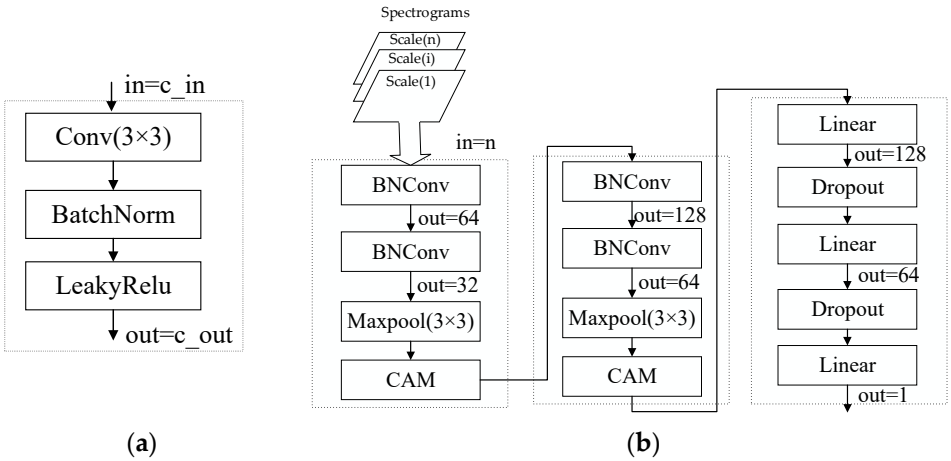


Figure 9. The proposed CNN architecture (**b**), in which BNConv is the block (**a**) that consists of a convolution layer, a Batch Normalization layer, and a Leaky ReLU.

4. Experiments and Results

In this section, the experimental results associated with the proposed methods are presented. First, we introduce the experimental setup and the datasets, and then present the results concerning the channel attention and the multi-scale channels.

4.1. Experimental Setup

We implemented the mentioned methods on the Pytorch platform, with the help of the Homura package [33]. We calculated the log-mel spectrogram according to the scale with 80 frequency bands, and with a hop size of 315 on the resampled signals with a sample rate of 22,050 Hz. The images were generated from the spectrogram with a height of 80, a width

of 150, and a hop size of 5, and then were fed to the CNN. Before training, we computed the min and max matrices, mean, and standard deviation in terms of the dataset. The datasets are described in the following section. Adam was utilized as the optimizer with the weight decay of 10^{-4}. The early stopping mechanism was also used, and the patience was set to 10. We employed the step-wise learning rate scheduler with a step size of 10. To ensure the effectiveness of the experimental comparisons, all of the parameters mentioned above were kept constant.

4.2. Datasets

The choice of dataset can lead to overfitting, in addition to optimistic results [11]. For example, a dataset with an insufficient number of samples would result in overfitting, and similar training and test datasets would lead to optimistic results. In this study, the aim of the experiments was mainly to test whether our method can improve the performance, rather than at proving that our proposed methods produce the best performance. In fact, our proposed methods are not exclusive and may be utilized in other CNNs, and even in RNNs. Therefore, the most important factor of the experiments was to ensure that the same conditions were maintained when applying our methods or not.

The experiments were also required to be performed on different datasets to prove the effectiveness of the proposed approaches. We chose three publicly available datasets for the experiments. The first dataset was the Jamendo corpus (JMD) [19,34], which contains 93 songs with a total length of 371 min. The second was RWC pop [35], which contains 100 songs with a total length of 407 min. The third was Mir1k [36], which is relatively smaller, having a total length of 133 min and comprising 1000 song clips with durations ranging from 4 to 13 s. The overall summary of the three datasets is shown in Table 2.

Table 2. Summary of the datasets used for the experiments.

Dataset	Number of Songs	Total Length (min)	Voice to Non-Voice Ratio
JMD	93	371	1.12
RWC	100	407	1.55
Mir1k	1000	133	4.37

The original division of the training, validation, and test datasets was unchanged for JMD [19], i.e., 61 songs for training, 16 for validation, and 16 for testing. We divided the RWC dataset according to the songs' ending digits, with 0–4 selected for the training dataset, 5 and 6 for the validation dataset, and 7–9 for the test dataset. We also separated the Mir1k dataset according to the songs' ending number characters: "05", "07", and "09" were chosen as the validation dataset; "06", "08", and "10" were chosen as the test dataset; and the remainder were assigned to the training dataset. Divided in this manner, the training, validation, and test datasets of RWC and Mir1k are stochastic to some extent. It is worth noting that Mir1k is not only small, but also highly unbalanced; however, this was not relevant because our aim was not to design a perfect dataset to obtain optimal results. The songs of JMD are all by different artists, whereas the 100 pop songs of RWC are performed by 34 singers, and the 1000 song clips of Mir1k are sung by 19 artists. Therefore, in terms of artists, the similarity between the training and the dataset is the most significant for Mir1k, and second-most significant for RWC; these similarities would lead to optimistic results to some extent. We expected that JMD would be the most challenging dataset for our task, whereas Mir1k would be the easiest.

4.3. Channel Attention

We first demonstrate the channel attention distribution with an example, and then present the results to prove the effectiveness of the channel attention method.

4.3.1. The Attention Distribution on the Feature Maps

Because interpretability is a notorious disadvantage of the DNN, it is not possible to clearly disclose how the CNN learns the importance of the feature maps. Thus, we illustrate an attention distribution example in Figure 10, and hope this helps explain its positive effects on the performance. As can be seen from Figure 10b, the features with some useful textures to discriminate the singing voice from the spectrogram were extracted by "observing" the upper layer feature maps. In the conventional CNNs, these feature maps are fed to the next layer with the same weight; that is, the attentions of the feature maps are imposed equally, and the corresponding gray level in Figure 10c is the same, i.e., all white. By comparison, after the attentions were measured by the scaled dot-product attention module, the importance of the feature maps was adjusted, and the next layer imposed specific attentions on those feature maps according to the attention distribution described in Figure 10c. The adjusted feature maps are shown in Figure 10d. It can be seen that some feature maps in the first column of Figure 10b are restrained because they may play a relatively unimportant role in detecting the singing voice.

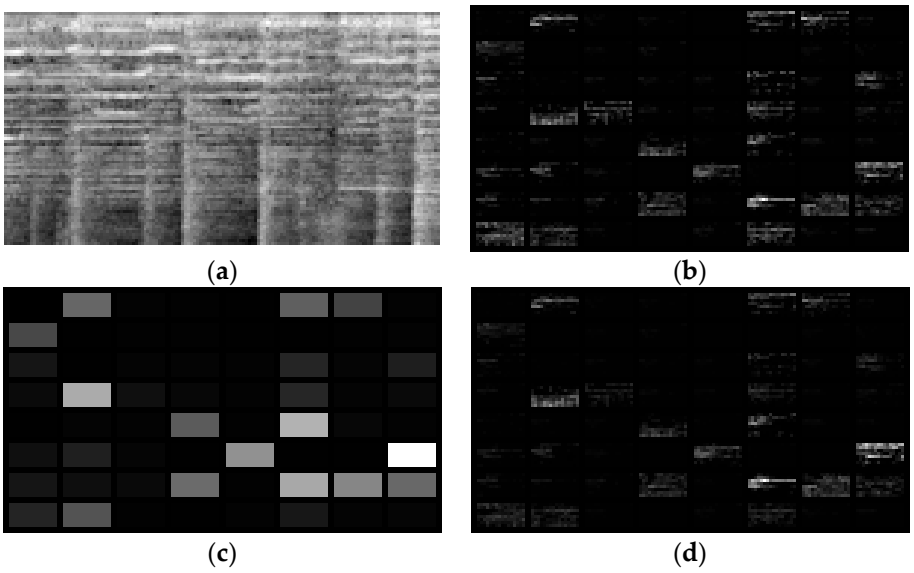


Figure 10. Comparison between the input and the output feature maps of a scaled dot-product attention module. (**a**) An input image from the log-mel spectrogram of the music "you are", (**b**) the input feature map, (**c**) the attentions or the weights of the corresponding channel of the output feature map (**d**).

4.3.2. The Experimental Results

In this section, the experimental results are presented to prove the effectiveness of our two proposed channel attention methods. To evaluate the results, we provide six metrics, i.e., Accuracy, F-measure, Precision, Recall, False Positive Rate (FPR) and False Negative Rate (FNR). Moreover, we provide the statistical rather than the optimal results because the CNN results are stochastic to some extent. The performance of the baseline architecture on JMD was reported in [22], in which Accuracy, F-measure, Precision, Recall, FPR, and FNR were 86.8, 86.3, 83.7, 89.1, 15.1, and 10.9 percent, respectively. However, the statistical results, for example, mean and standard deviation, were not presented. In order to provide more comprehensive performance results, we implemented the baseline architecture and the proposed architecture on the Pytorch platform, and conducted the experiments on the three datasets: JMD, RWC, and Mir1k. Furthermore, on each dataset, we trained, validated, and tested the three networks six times, with each of the scales (1024, 1536, 2048, 2560, 3072, and 3584) to achieve a better evaluation, i.e., run 36 times. Finally, we obtained the statistical results shown in Table 3, in which μ and σ refer to the mean and

standard deviation of the 36 results, respectively, and SDP-CNN refers to the scaled dot-product channel attention-based CNN, and SE-CNN refers to the squeeze-and-excitation channel attention-based CNN. From the table, there is very little difference in the mean performances on Jamendo for the baseline CNN between our implementation and the implementation of Lee et al. in [22], which indicates our implementation is believable.

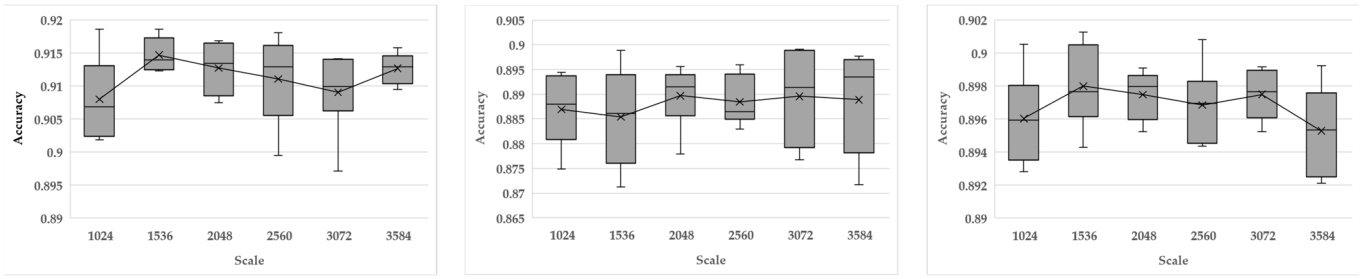
Table 3. The experimental results on the datasets: JMD, RWC, and Mir1k.

Dataset	Method	Accuracy	F-Measure	Precision	Recall	FPR	FNR
JMD	CNN ($\mu \pm \sigma$, %)	86.61 ± 1.01	86.35 ± 0.93	82.35 ± 2.37	90.9 ± 2.62	17.14 ± 3.1	9.1 ± 2.62
	SDP-CNN ($\mu \pm \sigma$, %)	88.78 ± 0.75	88.38 ± 0.68	85.55 ± 2.07	91.49 ± 1.99	13.58 ± 2.5	8.51 ± 1.99
	SE-CNN ($\mu \pm \sigma$, %)	88.82 ± 0.81	88.48 ± 0.72	85.21 ± 1.9	92.07 ± 1.56	14.02 ± 2.26	7.93 ± 1.56
RWC	CNN ($\mu \pm \sigma$, %)	88.85 ± 0.68	90.62 ± 0.63	91.43 ± 1.35	89.87 ± 1.84	12.67 ± 2.39	10.13 ± 1.84
	SDP-CNN ($\mu \pm \sigma$, %)	90.26 ± 0.6	91.75 ± 0.54	93.17 ± 1.09	90.54 ± 1.41	9.96 ± 1.82	9.59 ± 1.42
	SE-CNN ($\mu \pm \sigma$, %)	91.13 ± 0.52	92.49 ± 0.5	93.88 ± 0.92	91.17 ± 1.45	8.92 ± 1.54	8.83 ± 1.45
Mir1k	CNN ($\mu \pm \sigma$, %)	88.91 ± 0.23	93.73 ± 0.13	90.67 ± 0.43	97.01 ± 0.54	58.67 ± 3.31	2.99 ± 0.54
	SDP-CNN ($\mu \pm \sigma$, %)	89.42 ± 0.27	94 ± 0.16	91.14 ± 0.52	97.06 ± 0.64	55.43 ± 3.91	2.94 ± 0.64
	SE-CNN ($\mu \pm \sigma$, %)	89.67 ± 0.24	94.13 ± 0.14	91.47 ± 0.61	96.97 ± 0.72	53.17 ± 4.55	3.03 ± 0.72

It can be seen from the table that the proposed channel attention method improved the performances on all the datasets compared to the conventional CNN, whereas SE-CNN slightly outperformed the SDP-CNN. Moreover, the proposed two methods are more stable on the two bigger datasets, i.e., JMD and RWC, because they present the smaller variances. In terms of stability, the SDP-CNN is on par with the SE-CNN due to the insignificant differences in the variances of the metrics on the three datasets. In terms of F-measure, the most significant increase was achieved on JMD by SE-CNN, with 2.13 percent, whereas the smallest increase was on Mir1k by SDP-CNN, with 0.27 percent. In fact, the overall performance increase on Mir1k was the smallest among the three datasets. We think that Mir1k is the "easiest" dataset and its potential is almost completely exploited by the inferior method, i.e., the baseline CNN, so that the better methods cannot significantly improve the results. In addition, the FPR values for Mir1k are very high because of the unbalanced samples, even though we used the weighted binary cross entropy as the loss function.

4.4. Multi-Scale Channels

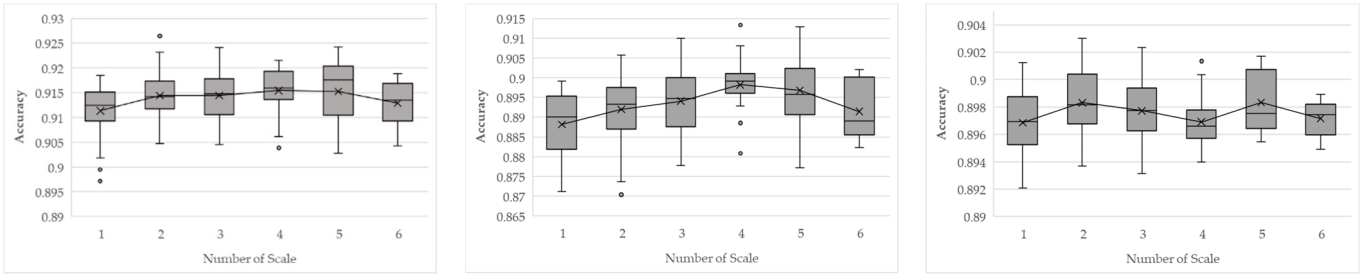
In this section, we demonstrate that the multi-scale channels strategy can improve the SVD performance and outperform the conventional single channel strategy. Before illustrating the multi-scale channels experimental results, we investigate the effects of the different scales of spectrograms on the performance. Figure 11 shows the boxplots of the accuracy performance on the three datasets with the different scales of spectrograms: 1024, 1536, 2048, 3072, and 3584. We described the experimental steps in the previous section. The reason for choosing these scales for the experiments is that their corresponding STFT windows are usually suitable for analyzing the music signal and can produce the reasonable results. It can be seen from the figure that the different scales can affect the accuracy performance to different degrees. On RWC, the scale of 1536 achieved the highest mean accuracy and was relatively stable. The scale of 2048 showed the best mean performance on JMD, and the scale of 1536 produced the best performance on Mir1k.



(**a**) (**b**) (**c**)

Figure 11. The accuracy performance on RWC (**a**), JMD (**b**), and Mir1k (**c**) with the different scales of spectrograms.

In addition to the single scale, we proposed that the multi-scale channels method can improve the overall performance with two or more scales of spectrograms. In order to validate our proposal, experiments using two, three, four, five, and six scales were conducted as follows. First, for each of the six numbers of scales, we created scale groups via combination. For example, the group containing two scales included the scales 1024 and 1536, 1024 and 2048, 1024 and 2560, etc. The total number of combinations in this group summed to $C_6^2 = 15$. Then, on each dataset, we ran the experimental procedures two times with each combination of the scales in each scale group with the SE-CNN. For instance, for the number of scales i, we ran the experiment $2C_6^i$ times on each dataset. Although the number of results differed in the number of scales, it was sufficient to achieve a statistical comparison. Finally, all the experimental results are shown using the boxplots of Figure 12. For the single scale, we present the experimental results mentioned in the previous section.



(**a**) (**b**) (**c**)

Figure 12. The accuracy performance on RWC (**a**), JMD (**b**), and Mir1k (**c**) with the different number of scales of the spectrogram.

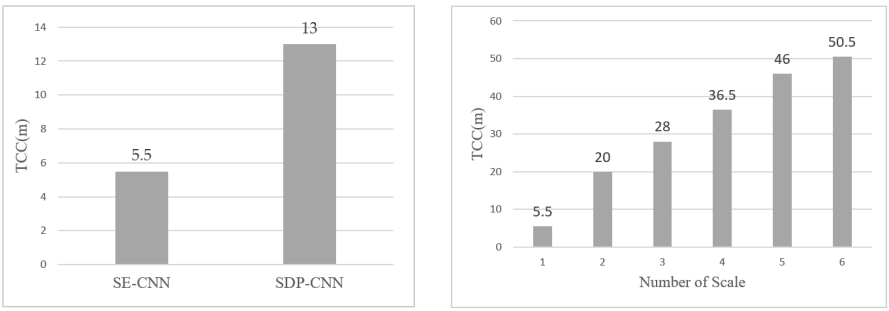
As shown in Figure 12, the mean accuracy performance was improved by our method. The significant increase happened on the most challenging dataset, JMD, in Figure 12b, in which the mean accuracy reaches the top for the number 4. On RWC, the mean accuracy also reached the top for the number 4, whereas the best performance was achieved with number 2 on Mir1k. Despite already being at a high level, the mean accuracy of the scale group of 4 outperformed the single scale group by 1% on JMD. We achieved a 0.4% increase at number 4 compared to the single scale on RWC. The most insignificant increase happened at number 2 on Mir1k because of the low potentiality, as previously discussed. Although, according to the experiments, we can conclude that the multi-scale channels can improve the SVD performance, we did not identify the best scale combination in this paper study because it was time consuming and we did not consider it to be necessary.

4.5. Computation Cost

In this section, we show the training computation cost (TCC) of the proposed methods. TCC depends on many factors, such as the network scale, the convergence mechanism,

the implementation method, and the server configuration; thus, we cannot measure it precisely.

We ran the SE-CNN and SDP-CNN with the single scale channel input on an Nvidia Tesla V100 with 32 GB memory and estimated TCC by minutes per training epoch on the JMD dataset. As shown in Figure 13a, SDP-CNN took significantly more time than SE-CNN, i.e., around 2.4-fold longer. This is because the computation of the attention needs the dot-product between the query and key matrices. As shown in Figure 13b, TCC increases when the number of scales of the combined spectrograms grows, but the growth rate is not exponential.



(**a**) (**b**)

Figure 13. TCC of the SE-SNN and the SDP-CNN (**a**), and the increase in TCC with the change in the number of scales (**b**).

5. Conclusions

In terms of CNN channels, we investigated two means to improve SVD performance and evaluated the results on three datasets. For channel attention, we proposed two methods: the scaled dot-product and squeeze-and-excitation, to measure the importance of the channels. It can be concluded from the related experimental results that the two channel attention mechanisms can be utilized for the CNNs to pay attention to the specific features in order to achieve better performance. In addition to raising the performance, the attention layer has the characteristic of "plug and play", which means the original CNN architecture does not need to be amended. In the proposed architecture, we only employed two attention layers. In the future, we will study whether the performance can be further improved by adding more attention layers. In addition, the experimental results showed that the FPR values were relatively high on the unbalanced dataset, Mir1k, despite using the weighted binary cross entropy as the loss function. Therefore, addressing the unbalanced data problem will be another research topic in the next stage.

For the multi-scale channels, we proposed to combine the multi-scale spectrograms and feed them to the CNN at the same time, from which the CNN can automatically find the appropriate characteristics from the different representations. The experimental results provided a positive performance improvement. Specifically, the accuracy of the four scales channel was increased by 1% on JMD compared to the previous relatively high level. Essentially, this invokes the question of which scale of the spectrogram can represent the original signal in the most appropriate manner. In this study, we proposed to answer this question using multi-scale input channels and by letting the CNN achieve the task by learning. However, a deterministic solution may perhaps be found if a relation can be established between the appropriate scale and the tempo of the song, because the durations of notes can determine the scale. Moreover, the original signal can be fed to the DNNs to achieve the appropriate representation by feature learning because the original signal can provide the information of all the scales. This approach is like that used in the end-to-end approach applied in the field of singing voice separation [37].

Finally, we expect that these methods can inspire and benefit the other tasks in the field of MIR, in addition to the field of speech recognition. Although these proposed

approaches are "plug and play" modules for the CNNs, the number of input channels should be increased in the multi-scale channels scenario.

Author Contributions: Conceptualization, W.G.; methodology, W.G.; software, W.G., X.Z. and Y.L.; validation, W.G., X.Z. and J.Z.; formal analysis, W.G. and J.Z.; investigation, W.G. and X.Z.; resources, Y.L.; data curation, W.G. and Y.L.; writing—original draft preparation, W.G.; writing—review and editing, Y.L. and J.Z.; visualization, W.G.; project administration, W.G.; funding acquisition, W.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Open Research Fund of Key Lab of Broadband Wireless Communication and Sensor Network Technology (Nanjing University of Posts and Telecommunications), Ministry of Education, and by the Chinese National Natural Science Foundation, grant number 61872199, and by the Jiangsu Overseas Visiting Scholar Program for University Prominent Young & Middle-aged Teachers and Presidents, and by the High Level Project of the Educational Cooperation between Jinling Institute of Technology, Jiangsu, China and Queensland University of Technology, Brisbane, Australia.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Jamendo corpus is available from http://www.jamendo.com (accessed on 1 May 2019); RWC pop is available from https://staff.aist.go.jp/m.goto/RWC-MDB/ (accessed on 1 May 2019); Mir1k is available from https://diana.shuu.cf/extdomains/sites.google.com/site/unvoicedsoundseparation/mir-1k (accessed on 1 March 2019).

Acknowledgments: The authors gratefully acknowledge the advices from Simon Dixon, Polina Proutskova, Daniel Stoller, Emir Demirel. They joined the discussion about the methodology and the experiments.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Chan, T.-S.; Yeh, T.-C.; Fan, Z.-C.; Chen, H.-W.; Su, L.; Yang, Y.-H.; Jang, R. Vocal activity informed singing voice separation with the ikala dataset. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brisbane, Australia, 19–24 April 2015; pp. 718–722.
2. Maddage, N.C.; Xu, C.; Wang, Y. Singer identification based on vocal and instrumental models. In Proceedings of the 17th International Conference on Pattern Recognition, Cambridge, UK, 23–26 August 2004; pp. 375–378.
3. Wang, Y.; Kan, M.-Y.; Nwe, T.L.; Shenoy, A.; Yin, J. Lyrically: Automatic synchronization of acoustic musical signals and textual lyrics. In Proceedings of the 12th Annual ACM International Conference on Multimedia, New York, NY, USA, 10–16 October 2004; pp. 212–219.
4. Berenzweig, A.L.; Ellis, D.P. Locating singing voice segments within music signals. In Proceedings of the IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics, New Platz, NY, USA, 24 October 2001; pp. 119–122.
5. Tsai, W.-H.; Wang, H.-M. Automatic singer recognition of popular music recordings via estimation and modeling of solo vocal signals. *IEEE Trans. Audio Speech Lang. Process.* **2005**, *14*, 330–341. [CrossRef]
6. Nwe, T.L.; Shenoy, A.; Wang, Y. Singing voice detection in popular music. In Proceedings of the 12th Annual ACM International Conference on Multimedia, New York, NY, USA, 10–16 October 2004; pp. 324–327.
7. Maddage, N.C.; Xu, C.; Wang, Y. A svm-based classification approach to musical audio. In Proceedings of the International Society for Music Information Retrieval (ISMIR), Baltimore, MD, USA, 27–30 October 2003.
8. Lehner, B.; Widmer, G.; Sonnleitner, R. On the reduction of false positives in singing voice detection. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Florence, Italy, 4–9 May 2014; pp. 7480–7484.
9. Schlüter, J. Learning to pinpoint singing voice from weakly labeled examples. In Proceedings of the International Society for Music Information Retrieval (ISMIR), New York, NY, USA, 7–11 August 2016; pp. 44–50.
10. Schlüter, J.; Grill, T. Exploring data augmentation for improved singing voice detection with neural networks. In Proceedings of the International Society for Music Information Retrieval (ISMIR), Malaga, Spain, 26–30 October 2015; pp. 121–126.
11. Lehner, B.; Schlüter, J.; Widmer, G. Online, loudness-invariant vocal detection in mixed music signals. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2018**, *26*, 1369–1380. [CrossRef]
12. Huang, H.-M.; Chen, W.-K.; Liu, C.-H.; You, S.D. Singing voice detection based on convolutional neural networks. In Proceedings of the 7th International Symposium on Next Generation Electronics (ISNE), Taipei, Taiwan, 7–9 May 2018; pp. 1–4.
13. Schlüter, J.; Lehner, B. Zero-mean convolutions for level-invariant singing voice detection. In Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR 2018), Paris, France, 23–27 September 2018; pp. 321–326.

14. Lehner, B.; Widmer, G.; Böck, S. A low-latency, real-time-capable singing voice detection method with lstm recurrent neural networks. In Proceedings of the 23rd European Signal Processing Conference (EUSIPCO), Nice, France, 31 August–4 September 2015; pp. 21–25.
15. Leglaive, S.; Hennequin, R.; Badeau, R. Singing voice detection with deep recurrent neural networks. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brisbane, Australia, 19–24 April 2015; pp. 121–125.
16. Zhang, X.; Yu, Y.; Gao, Y.; Chen, X.; Li, W. Research on singing voice detection based on a long-term recurrent convolutional network with vocal separation and temporal smoothing. *Electronics* **2020**, *9*, 1458–1470. [CrossRef]
17. Berenzweig, A.; Ellis, D.P.; Lawrence, S. Using voice segments to improve artist classification of music. In Proceedings of the 22nd Audio Engineering Society International Conference, Espoo, Finland, 15–17 June 2002.
18. Li, Y.; Wang, D. Separation of singing voice from music accompaniment for monaural recordings. *IEEE Trans. Audio Speech Lang. Process.* **2007**, *15*, 1475–1487. [CrossRef]
19. Ramona, M.; Richard, G.; David, B. Vocal detection in music with support vector machines. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Las Vegas, NV, USA, 31 March–4 April 2008; pp. 1885–1888.
20. Regnier, L.; Peeters, G. Singing voice detection in music tracks using direct voice vibrato detection. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Taipei, Taiwan, 19–24 April 2009; pp. 1685–1688.
21. Mauch, M.; Fujihara, H.; Yoshii, K.; Goto, M. Timbre and melody features for the recognition of vocal activity and instrumental solos in polyphonic music. In Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR), Miami, FL, USA, 24–28 October 2011; pp. 233–238.
22. Lee, K.; Choi, K.; Nam, J. Revisiting singing voice detection: A quantitative review and the future outlook, International Society for Music Information Retrieval Conference. In Proceedings of the International Society for Music Information Retrieval, Paris, France, 23–27 September 2018.
23. Graves, A.; Wayne, G.; Danihelka, I. Neural turing machines. *arXiv* **2014**, arXiv:1410.5401.
24. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* **2014**, arXiv:1409.0473.
25. Luong, M.-T.; Pham, H.; Manning, C.D. Effective approaches to attention-based neural machine translation. *arXiv* **2015**, arXiv:1508.04025.
26. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
27. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7132–7141.
28. Hu, Y.; Li, J.; Huang, Y.; Gao, X. Channel-wise and spatial feature modulation network for single image super-resolution. *IEEE Trans. Circuits Syst. Video Technol.* **2019**, *30*, 3911–3927. [CrossRef]
29. Chen, L.; Zhang, H.; Xiao, J.; Nie, L.; Shao, J.; Liu, W.; Chua, T. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning, computer vision and pattern recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6298–6306.
30. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
31. Maas, A.L.; Hannun, A.Y.; Ng, A.Y. Rectifier nonlinearities improve neural network acoustic models. In Proceedings of the International Conference on Machine Learning (ICML) Workshop on Deep Learning for Audio, Speech, and Language Processing, Atlanta, GA, USA, 16 June 2013; pp. 3–11.
32. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference on Machine Learning, PMLR, Lille, France, 7–9 July 2015; pp. 448–456.
33. Homura. Homura Package. Available online: https://github.com/moskomule/homura (accessed on 10 October 2019).
34. Grard, P.; Kratz, L.; Zimmer, S. Jamendo, Open Your Ears. Available online: http://www.jamendo.com (accessed on 1 May 2019).
35. Goto, M.; Hashiguchi, H.; Nishimura, T.; Oka, R. Rwc music database: Popular, classical and jazz music databases. In Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR), Paris, France, 13–17 October 2002; pp. 287–288.
36. Hsu, C.-L.; Jang, J.-S.R. Mir-1k Dataset. Available online: https://diana.shuu.cf/extdomains/sites.google.com/site/unvoicedsoundseparation/mir-1k (accessed on 1 March 2019).
37. Stoller, D.; Ewert, S.; Dixon, S. Wave-u-net: A multi-scale neural network for end-to-end audio source separation. In Proceedings of the 19th International Society for Music Information Retrieval Conference, Paris, France, 23–27 September 2018; pp. 334–340.