Human-centric Computing and Information Sciences

RESEARCH Open Access

Comparative study of singing voice detection based on deep neural networks and ensemble learning

Shingchern D. You*, Chien-Hung Liu and Woei-Kae Chen

*Correspondence: you@csie.ntut.edu.tw
Department of Computer Science and Information Engineering, National Taipei University of Technology, Taipei, Taiwan

Abstract

This paper investigates various structures of neural network models and various types of stacked ensembles for singing voice detection. The studied models include convolutional neural networks (CNN), long short term memory (LSTM) model, convolutional LSTM model, and capsule net. The input features to the network models are MFCC (mel-frequency cepstrum coefficients), spectrogram from short-time Fourier transformation, or raw PCM samples. The simulation results show that CNN model with spectrogram inputs yields higher detection accuracy, up to 91.8% for Jamendo dataset. Among the studied stacked ensemble methods, performing voting strategy yields comparable performance as the other methods, but with much lower computational cost. By voting with five models, the accuracy reaches 94.2% for Jamendo dataset.

Keywords: Sining voice detection, DFT, MFCC, Convolutional neural networks

Introduction

Detecting signing voice in a piece of music work (soundtrack) has been studied for many years because this technique is the foundation for many advanced applications [1]. In the following, we briefly describe some of the applications. Firstly, if we intend to remove the vocal sound from a singing soundtrack for karaoke singers, the pre-processing step certainly needs to pin point the audio segments with singing voice [2]. Second, we know that the most well-known portion of a western popular song is usually on the verse part, which almost always contains singing performance. Therefore, the work of music summarization [3] as well as melody extraction [4] can also benefit from knowing the segments with signing voice. Next, if we want to identify the singer in a music work, we need to have the singing segments before conducting recognition [5]. In addition to the above applications, if we intend to perform a lyrics-to-melody conversion [6], we also need to know the signing segments. From the above examples, we know that singing voice detection is a fundamental pre-processing step for many applications.

There are two types of problems in detecting singing voices in a piece of audio work. The first type is to mark the starting and ending points of all vocal segments on the soundtrack, referred to as the singing voice segmentation problem. The second type is to determine whether a short audio clip (e.g., 2 s) contains any human-perceivable vocal sound, including the vocal sound of the background vocalists. This type of problem is referred to as the singing voice detection problem. At a first thought, it seems that the singing voice segmentation problem is more difficult than the detection problem. However, the segmentation problem has the entire soundtrack available; therefore, some post-processing steps, such as smoothing [7], could be conducted to correct some error labeling by using the information of the preceding and succeeding pieces of music. Such a technique, however, could not be applied to the singing voice detection problem. In this paper, we focus on the second type of problem. As these two types of problems are somewhat different, the accuracy measure reported for one type of problem may not be compared with that of the other.

Recently, new types of neural network structures have been widely applied to many difficult problems [8, 9]. For example, the CNN structure nowadays is widely used in the image recognition problems [10, 11]. Therefore, it is a natural extension to apply the CNN structure to the singing voice detection problem. Conceptually, if we construct the spectral-temporal features of an audio clip as a two-dimensional feature plane, we should be able to use CNN structures originally proposed for images in this problem. One well-known spectral-temporal feature is the MFCC (mel-frequency cepstrum coefficients) [12]. However, there are also other types of spectral-temporal features available, such as the spectrogram based on STFT (short-time Fourier transformation). In addition, directly applying raw PCM (pulse code modulation) samples to a CNN-based structure [13] is also a possible alternative. With all these possibilities, it is useful to know which type of features yields better accuracy. To have a fair and meaningful comparison, preferably the classifiers are all based on CNN and have similar structures. This issue is the first subject we would like to investigate in this paper.

As there are many different types of neural networks available in the literature, it is also important to investigate whether one particular type of neural networks has higher accuracy than others. For example, it is known that the LSTM-based neural network is also effective for audio genre classification problems [14]. Thus, LSTM may also perform well in singing voice detection. In addition to the conventional LSTM, we also investigate one of LSTM variations, called convolutional LSTM [15], which combines the convolution layer into the LSTM structure to adapt to both spacial and temporal features. This network structure may also be a good candidate for singing voice detection. In addition, we also attempt to use the capsule network (capsulenet) [16] to this problem. Overall, we will report the results of using CNN, LSTM, convolutional LSTM, and capsulenet in this paper.

Finally, it is well-known that the ensemble learning can improve the detection accuracy in many instances. Thus, we would like to study if this approach can still work for singing voice detection problem. In this paper, we apply the ensemble learning with three different approaches, namely, voting, post-classifier, and fusion for singing voice detection problem and report the respective accuracy. Overall, the goal of this paper is to present a comprehensive comparison of relative accuracy performance among various types of features, network models, and ensemble learning techniques for the singing voice detection problem, so that researchers and practicing engineers facing this problem could follow our findings without repeating all sorts of experiments again.

This paper is organized as follows. "Related work" section is the literature survey, covering related papers with the reported accuracy. "Neural network structures with various types of features" section describes all of the network structures and ensemble learning approaches to be used in the experiments with various spectral-temporal features. "Experiments and results" section covers experimental setting, the used datasets, and the experimental results. Finally, "Conclusion" section is the conclusion.

Related work

To locate singing voice segments, researchers usually extract one or more types of features from the audio signal and then use a classifier for detection. One widely used feature for audio applications is the MFCC (Mel Frequency Cepstral Coefficient). To investigate whether this type of feature is better than others, Kim et al. [17] compared MFCC with MPEG-7 ASP (Audio Spectrum Projection) features, and found that MFCC was better. Similarly, Rocamora and Herrera [18] had the same finding, but their accuracy was only around 78%. To further increase the accuracy, Dittmar et al. [19] proposed to combine MFCC features with vocal variation and Flutogram variation. When using random forest as the classifier, the F-measure could reach 87%.

Other than the MFCC features, Berenzweig and Ellis [6] used the statistical features and the HMM (Hidden Markov Model) as the classifier. They reported an accuracy of around 80%, indicating that the statistical features may not be much better than the MFCC feature.

Another attempt to improve the accuracy for singing voice segmentation problem is through the post-processing step. To this end, Lukashevich et al. [7] proposed to use the ARMA (autoregressive moving average) smoothing model as the post-processor. With this technique, they had an average accuracy of 82.5%. Vembu and Baumann [20] also combined several features with a smoothing technique. They yielded an accuracy of 84% for singing voice segmentation.

Nwe et al. [21] proposed the bootstrapping technique to further improve the accuracy of singing voice segmentation. They incorporated the musical features and musical structure as features and used a Multi-Model HMM as the classifier. With the bootstrapping technique, they had an accuracy of 86.7%.

In terms of classifiers, Leglaive et al. [22] compared a neural network model called BLSTM (Bidirectional Long Short-term Memory) with traditional classifiers, such as SVM (support vector machine) for singing voice segmentation. In their setting, the features are MFCC-like features derived from two HPSS (Harmonic/Percussive Source Separation) layers. According to their simulations, BLSTM was better, with the F-measure reaching 91%.

On the singing voice detection problem, Schluter and Grill [23] proposed a model using three-layer convolutional neural networks (CNN) for signing voice detection. The features to the CNN are 2-D spectral-temporal feature plane, obtained with a procedure similar to that of the MFCC. When applying data augmentation, they reached an accuracy of 91%.

To remove the need of feature extraction for singing voice detection, Dieleman and Schrauwen [13] used a unified network for both feature extraction and classification. Hypothetically, using a learnable network for feature extraction should be able to extract better features than existing ones, as suggested by Humphrey et al. [24]. However, simulation results in Dieleman and Schrauwen's report showed that this type of unified networks did not provide higher accuracy when compared with networks using traditional features, such as MFCC. Recently, Lee et al. proposed a new end-to-end network model [25] with many layers of small filters. The authors claimed that their model was better than the conventional end-to-end model given in [13]. As their experiments did not cover singing voice detection, whether this structure is better in this particular problem is not concluded.

Neural network structures with various types of features

This section describes all network structures studied in this paper. To have a fair comparison, we try our best either to use the same type of features for some network structures, or similar network structures for different types of features.

CNN with MFCC features

Figure 1 shows the CNN structure for the MFCC feature, called MCNN in the following. In fact, using MFCC-like features followed by a CNN is widely applied to various audio classification problems, such as audio events [26]. In our structure, the input is an MFCC plane with a size of 80×63, where 80 denotes the number of MFCC bands, excluding DC, and 63 is the number of MFCC frames in the audio clip. The number of MFCC bands and the window hop size for successive bands have previously been determined by experiments [27]. In the MFCC structure, the first convolution layer has 16 kernels with a size of 4×3. Thus, after this layer, we have 16 internal planes with each one having a size of 77×61 without using zero padding. In our previous study [27], we used a convolution kernel of 3×3. But we later found that a kernel of 4×3 had slightly better accuracy. Please note that Wu et al. [28] also reported that non-square kernels provided slightly better accuracy in audio applications. The second layer is again a convolutional layer with 16 kernels sized 4×3, producing planes with a size of 74×59. In the convolutional layers, the activation function is ReLU (Rectified Linear Unit). The second convolutional layer is followed by a pooling layer to pick the maximum value within a scope of 2×2. After the third convolutional layer and the second pooling layer, we use 128 sets of $17 \times 13 \times 16$ filters to convert the 2-D polling layer into a one-dimensional structure. Finally, this one-dimensional layer connects two output nodes. The activation function for the one-dimensional layer is a sigmoid function, and the output layer uses the softmax function.

CNN with FFT features

The second type of the feature plane is the spectrogram [29]. In this type of features, the incoming audio signal is multiplied by a hamming window. The window length was

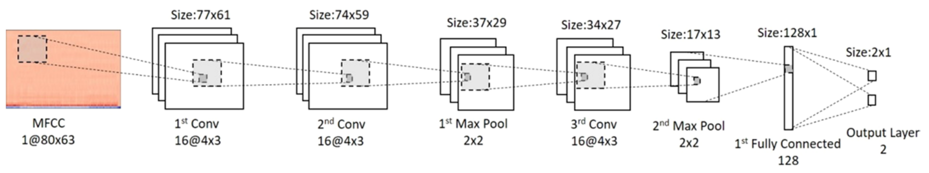

Fig. 1 MCNN structure

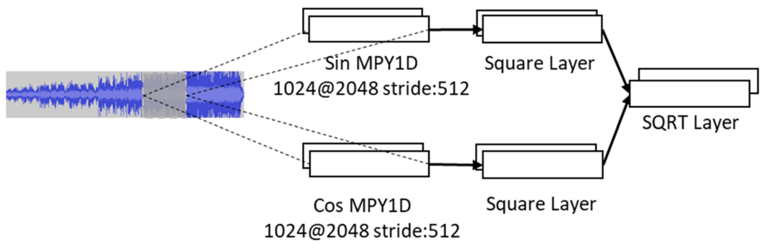

Fig. 2 Network structure to compute spectrogram

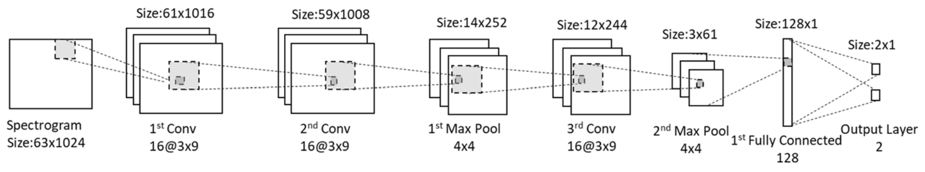

Fig. 3 SCNN structure. The input is a spectrogram

experimentally determined as 2048 and the hop size as 512. The windowed signal is then transformed to spectral domain by a short-time Fourier transformation (STFT). As the STFT coefficients are complex-valued, we take their modulus before sending to CNN for processing. In order to be realizable in network structure, the windowed signal is multiplied by weights containing sine and cosine coefficients. There are 1024 sets of weights, with each one having 2048 coefficients, as shown in Fig. 2. At the output of the SQRT (square root) layer, we obtain a spectrogram of size of 63×1024, where 1024 represents the frequency bins and 63 represents the time instances.

In Fig. 2, we use the square layer to take squares of the outputs from sin MYP1D and cos MYP1D, and then take the squared roots of the added values. The reason of taking square is to avoid producing negative values. Basically, point wise convolving the input signal with a sine or cosine function may produce either a positive or a negative number, depending on the phase of the input signal. In our case, we are not concerned with the phase of the signal, but only the relative "strength" (energy) of the signal. Therefore, we use the square function. Actually, we have tried removing the square and the square-root functions in the experiments, but the accuracy with such an arrangement was much lower.

The CNN structure to host the spectrogram is shown in Fig. 3. Although similar to Fig. 1, due to the big size of the input plane, the sizes of the convolution and pooling kernels are enlarged. To have a fair comparison, however, we do not increase the number of layers or change the sequence of the layers. In the following, this structure is called SCNN indicating that it is a CNN structure with inputs derived from STFT.

CNN with raw PCM

The third case in our investigation is applying PCM samples to the CNN without any pre-processing steps [27], also known as end-to-end processing, as proposed in [13]. In this paper, to be able to perform a meaningful and fair comparison, we still use a general

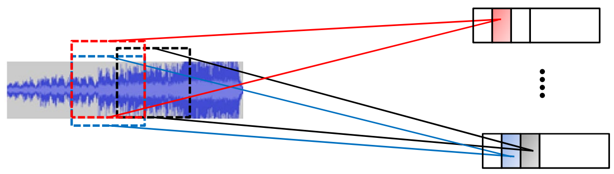
Fig. 4 Mapping from PCM samples to internal 2-D representation [27]

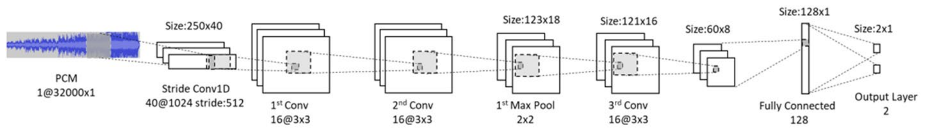

Fig. 5 End-to-end CNN structure [27]

structure similar to Figs. 1 and 3. To this end, the PCM samples in the audio clip are converted into an internal 2-D representation through 40 filters, as shown in Fig. 4, where each filter has 1024 coefficients and the coefficients are obtained by learning instead of designing. To fit into the general CNN structure, we use a stride length of 512. That is, the distance between successive multiplication-add (MLA) operations is 512 samples instead of one sample in the conventional convolution. In Fig. 4, two black dashed rectangles represent the covered portions of successive MLA operations and produce two real values on the left. The red dashed window indicates a different filter in use to produce another series of filtered values. In our setting, each filter produces 250 values. So, the feature plane has a size of 250×40.

The end-to-end CNN is shown in Fig. 5. To reflect the actual connections, Fig. 4 is included in Fig. 5 for completeness. In this CNN, the general structure is similar to the previous two for fair comparison. To save space, these meta-parameters are given in the figure without additional explanation. In the following, this structure is called ECNN.

LSTM for singing voice detection

We mentioned in "Related work" section that the LSTM network has been applied to singing voice segmentation problem with satisfactory results [22]. Thus, we also include this structure in the comparison. As the LSTM is widely used nowadays, we omit its general description here to save space and only concentrate on its usage. The LSTM structure used in the experiment is given in Fig. 6. The inputs to the LSTM network are slices of the spectrogram. That is because the sliced spectrogram feature yields slightly better accuracy than that of the MFCC used for CNN structure, to be discussed in "Experiments and results" section. Overall, 63 sets of coefficients are sent to the network during training before resetting the memory cells. In the network, the first layer is an LSTM layer with 1024 inputs and 128 nodes (internal outputs). The second layer, having also 128 nodes, takes the outputs from the first layer. Then, the 128 s-layer outputs connect to fully connected layers for decision making. In the following, this structure is called SLSTM.

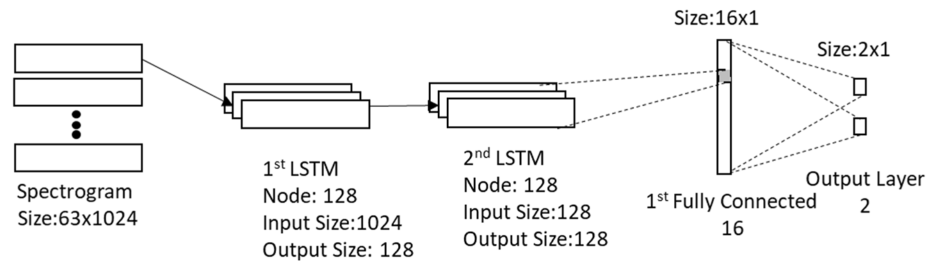

Fig. 6 LSTM network with sliced spectrogram as inputs

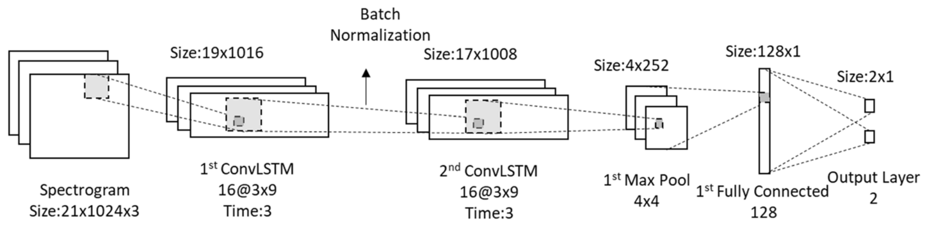

Fig. 7 Initial CLSMT network for trials

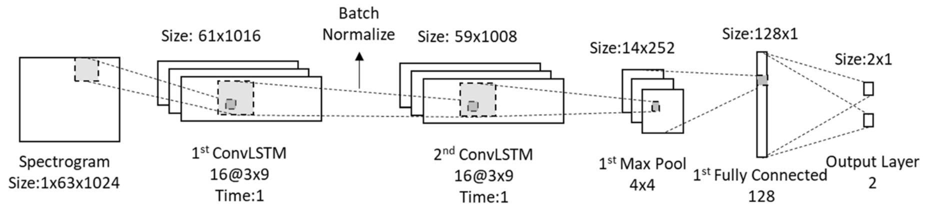

Fig. 8 Final CLSMT network used in the experiments

Convolutional LSTM

As the LSTM network only accepts one-dimensional arrays as inputs, the convolutional layer could not be utilized. To take advantage of the convolutional layer, the convolutional LSTM was proposed [15]. In the following, we refer the convolutional LSTM as CLSTM. As the CLSTM takes several two-dimensional planes as inputs, conceptually it could perform better than the traditional LSTM network with the spectrogram inputs. In our initial setting, the structure of the CLSTM network is depicted in Fig. 7, where the entire spectrogram for the 2 s audio is divided into three consecutive sub-planes as the input sequence. To speed up the training, batch normalization is utilized between convolutional LSTM layers. However, we find that the accuracy of using three sub-planes is almost the same as keeping the entire spectrogram as one plane to the network, as shown in Fig. 8. Therefore, we decide to keep this structure and report its accuracy. In this regard, the main difference between the CLSTM and SCNN in the experiments mainly lies on the type of processing nodes (ReLU vs LSTM) in the convolutional layers.

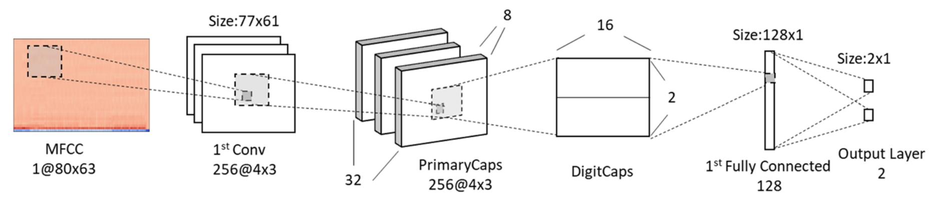

Fig. 9 The capsule net used in the experiments

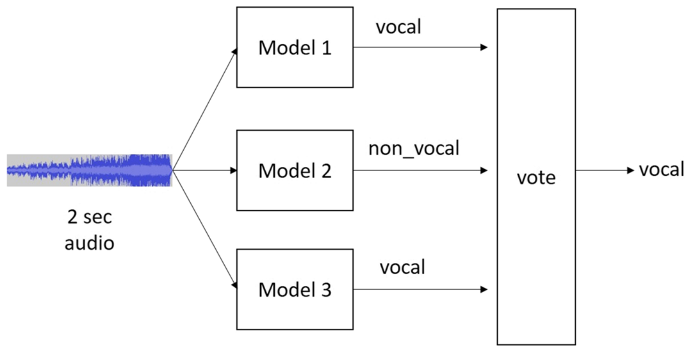

Fig. 10 Illustration of voting as a type of stacked ensembles

Capsule network

Sabour et al. proposed a new type of network, called capsule net, in 2017 [16]. The unique feature of the capsule network is that the internal representations are arranged in vectors, instead of scalars. In their paper, they showed some advantages of using this type of network. To see if this type of network is suitable for singing voice detection, we followed the structure in their paper with necessary modifications in hyper-parameters. The capsule net used in the experiments is shown in Fig. 9. In this structure, due to the large amount of processing units, we are unable to use the spectrogram as the input plane. Therefore, we use the MFCC plane as the input instead. In our implementation, however, we do not implement the reconstruction network, as Sabour et al. did, because it is much more difficult to reconstruct a real-valued plane than a binary (black-and-white) plane in a simple fully connected network. To save space, we label all hyper-parameters in the figure without additional explanations. The interested reader should refer to the original paper [16] discussing the capsule net for details.

Ensemble learning by stacking

It is known that ensemble learning generally can improve the accuracy for many prediction (classification) problems. There are many different types of ensembles for machine learning. In this paper, we consider only the stacking type of ensemble learning. In this subsection, we briefly describe the used methods in the experiments.

Voting

The simplest stacking type of ensembles is voting. For example, if we have trained three classifiers, we can use a simple voting to determine the final decision, as shown in Fig. 10.

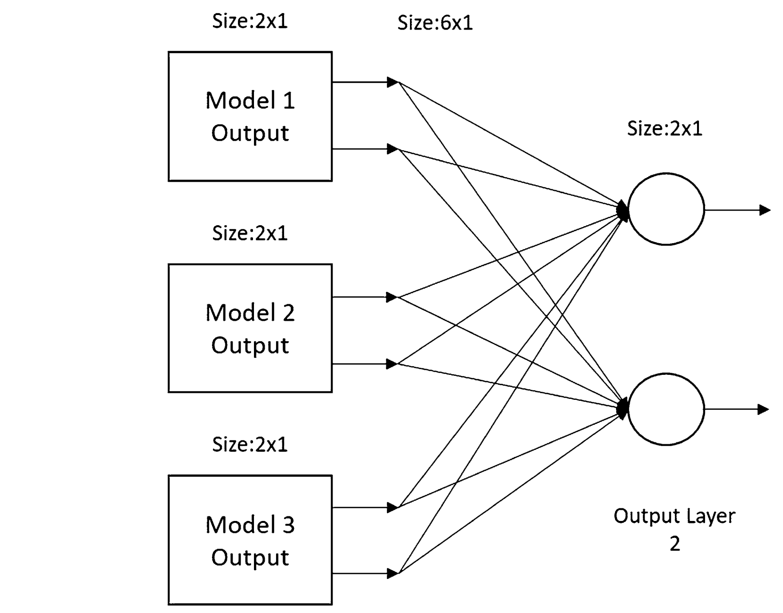

Fig. 11 A post classifier for stacking type of ensembles

In the experiments, we will use the decisions of either five or three classifiers as basis to vote and present the accuracy.

Post-classifier

The voting mechanism can be considered as a degenerated type of post classifier with equal weights for each binary classifier output. In a more general form, the weight for each classifier can be different to account for different capabilities of the classifiers. To do so, we use another simple neural network with the sigmoid activation function as the post-classifier, as shown in Fig. 11. Note that it is also possible to include an additional intermediate (hidden) layer in front of the post-classifier output layer. However, our simulation results show that this additional hidden layer does not improve the accuracy. Thus, we only report the accuracy results using the arrangement in Fig. 11.

Fusion

The post-classifier approach, in a sense, can be further generalized to adapt weights in each individual classifier in addition to weights in the post-classifier. Doing so could possibly achieve a higher level of optimization, hopefully leading to better accuracy in the stacked decision. To this end, we combine several network models together as a mega network, as shown in Fig. 12, and use the same training algorithm (back propagation) to train the weights of this mega network. It can be readily recognized from Fig. 12 that the top network model is MCNN, the middle one is SCNN, and the bottom one is CLSTM. Conceptually, this approach should yield better accuracy. However, we have the convergence problem during training, and thus this approach may not be so much useful in practice. The details are given in the experiments ("Experiments and results" section).

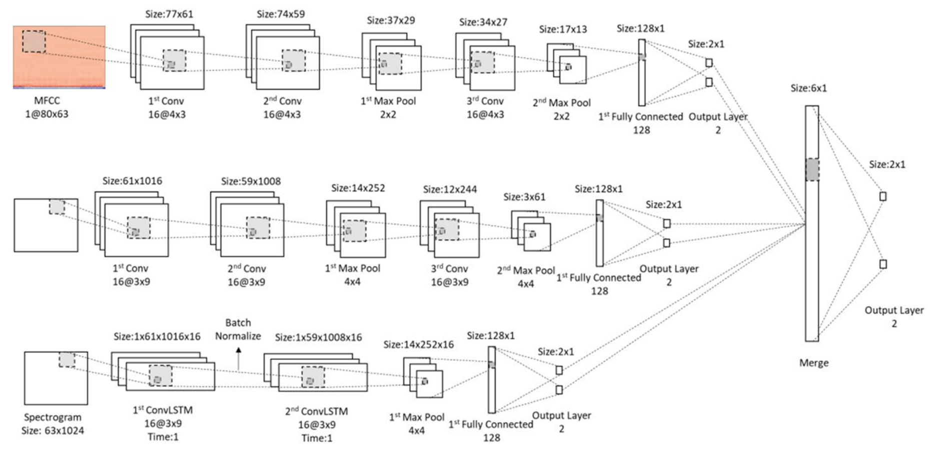

Fig. 12 Fusion type of stacked ensembles

Experiments and results

This section covers the experimental procedures and results. To ease reading, we divide the experimental results into two subsections, one for accuracy for each type of network models, and the other one for results using stacked ensembles.

Experimental datasets

As mentioned previously, each sample used in the experiments is an audio clip with duration of 2 s. The network models are trained to detect whether any vocal sound (singing voice) is present in the clip or not. Note that we consider that as long as the presence of vocal signal is distinguishable, it could only be in a portion of the audio clip. Furthermore, audio clips containing vocal sounds from backing vocalists are also classified as vocal clips. In the experiments, the ground truth (vocal or non-vocal status) of the samples is annotated by human listeners.

In the experiments, we use two datasets to assess the accuracy. The first dataset is the Jamendo dataset [30], containing 93 soundtracks, and equivalent to 6 h of playing time. The soundtracks are divided into training, validation, and testing sets, each with 61, 16, and 16 soundtracks. One advantage of the Jamendo dataset is that the singing segments in each soundtrack have been manually annotated. Therefore, all we need to do is to partition each soundtrack into many 2 s audio clips. In the experiments, the training audio clips are from both the training and validation sets, leaving only the testing set for testing audio clips. Overall, we have about 13,000 training samples and 3000 testing samples. The training set has 8480 vocal segments and 7868 non-vocal segments, and the test set has 1487 vocal segments and 1499 non-vocal segments.

The second dataset is derived from the FMA (free music archive) website [31]. The site contains more than 100,000 contributed soundtracks. In the experiments, we randomly pick about 18,000 soundtracks covering all types of music genres. Since the FMA dataset is not balanced in terms of music genre, the genre types of Rock, Electronic, and Experimental (out of 20 + genres specified in FMA website) cover about 60% of the chosen soundtracks.

For each soundtrack, we randomly take a 2-s excerpt as one sample for experiments. In this arrangement, no two audio clips in this dataset are from the same soundtrack. As the FMA dataset does not contain any annotation about the vocal/non-vocal information, we obtain the ground truth by human listeners. Specifically, we have more than 10 graduate students involved in the listening work. Each student is given around 1500 excerpted (2 s) segments. After listening to a segment, he/she is asked to identify whether the segment contains vocal or not. If in doubt, he/she is allowed to listen to the segment repeatedly. If the listener still cannot determine whether the segment is vocal or not, this segment is labeled as "undetermined," and it is not used in the experiments. With such a procedure, we finally have 4783 vocal segments and 7451 non-vocal segments in the training set, and 1660 vocal segments and 2485 non-vocal segments in the testing set. The annotated dataset is available in [32].

Experimental setting and environment

The experiments are conducted on top of the Tensorflow framework [33], with the use of the Keras library [34]. To increase the processing speed, we use GPUs (graphic processing unit) for training and testing computation. The experimental environment is listed in Table 1. The Tensorflow framework provides supports to efficiently use GPUs for training and testing. Therefore, we do not need to write low-level code, such CUDA, to use the GPUs. To further reduce the training time, we use three graphics cards and split the training work evenly into these cards. The training time for some of the network models used in the experiments can be accomplished within a couple hours using three graphic cards for 200 training epochs. Other network models do require substantially longer training time, up to several days.

In terms of training, we use the back propagation algorithm with ADADELTA [35] to adapt the learning rate. In addition, to avoid overfitting, we also use dropout regularization [36] for all layers, except the output layer, in the CNN structures. The probability for dropout is set to 0.5.

The format of the source audio clips is stereo audio with a sample rate of 44.1 ks/s. The pre-processing steps for the clips include down-mixing to one channel (mono) and downsampling to 16 ks/s. After the pre-processing, different types of features are extracted according to the used network models given previously.

During experiments, we obtain the accuracy for each model with respect to the number of training epochs, up to 200. Then, the reported accuracy is the highest

Table 1 Experimental environment

Component	Model
CPU	i7-6850k
Motherboard	x99T-ws
Memory	32G
Graphics card	1080ti × 3
OS	Ubuntu 16.04
Tensorflow	v1.0.1
Keras	2.1.2

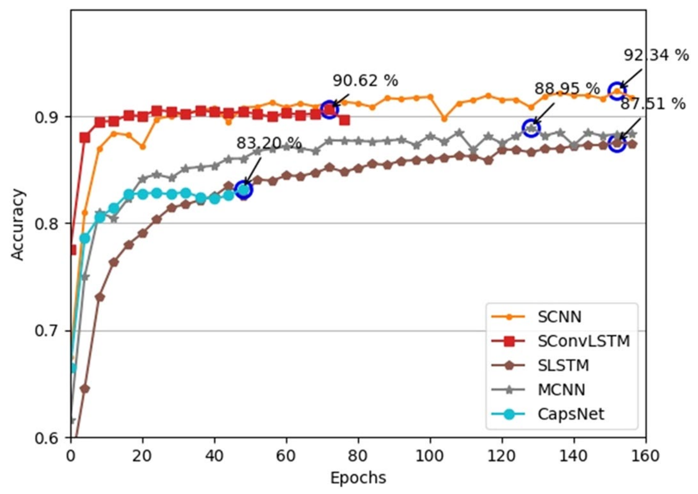

Fig. 13 The determination of the accuracy in one trail

Table 2 Accuracy for various network models trained and tested using the Jamendo dataset

Network model	Average accuracy (10 trials) (%)
MCNN	88.2
SCNN	91.8
ECNN	77.1
SLSTM	87.7
CLSTM	90.4
Capsule net	82.5

accuracy observed during training, as shown in Fig. 13. Since the accuracy of a model varies with re-initialization (i.e., assigning a new set of initial weights), we report the average accuracy based on 10 trails (10 training/testing processes with re-initialization) to reduce the accuracy fluctuation.

Experimental results for various network models

This section contains experiments intend to (a) compare the detection accuracy for various types of models given in the previous section, (b) observe the accuracy influenced by the unequal number of segments in vocal and non-vocal classes, and (c) evaluate the generalization capability of the studied networks when trained in one dataset and tested in another dataset. In this experiment, the training and testing samples are either from the Jamendo dataset or from the FMA dataset.

As a baseline, the network models mentioned previously are trained and tested using the Jamendo dataset. The accuracy results are given in Table 2. From Table 2 we know that SCNN has higher accuracy. On the other hand, the end-to-end model (ECNN) yields very low accuracy and also required extremely long training time.

One possible reason for the low accuracy for the ECNN model is due to overfitting, because the highest accuracy appears after 20 or 30 epochs and then the accuracy declines with more training epochs [27]. As the number of weights in this network model is extremely large, it seems that overfitting is inevitable without significantly increasing the size of the training set. However, as the accuracy of this model is more than 10 percent lower than other models, we would need at least four to eight times more training data to successfully train this model. Having running out of available training data, we are unable to improve the accuracy of the ECNN model.

As mentioned in "Related work" section, a new deep network structure for end-to-end audio recognition with many small filters was proposed [25]. The authors showed that their model was better than the conventional end-to-end model given in Fig. 5. When using this deep structure in the experiments, the average accuracy did increase from 77% to around 82%. But, it is still significantly lower than that of the SCNN structure. Due to the low accuracy and prolonged training time, we decide not to further investigate the end-to-end approach in the following experiments.

Other than the ECNN, we also notice that the capsule net has relatively lower accuracy. This phenomenon could be partially due to sub-optimal hyper-parameters of the network model. We have also tried several different sets of hyper-parameters; unfortunately, we are still unable to find a good set of hyper-parameters for this particular type of model. We still report our results, hoping that other researchers may benefit from our findings.

When comparing the SLSTM and CLSTM models, we notice that CLSTM is slightly better. This result seems to indicate that using convolutional layers is beneficial for 2-D feature planes, such as spectrograms.

Since the MCCN and SCNN have similar network structures, the performance difference is mainly due to the type of input features. Generally speaking, the spectrogram is a lower level of spectral-temporal features than that of the MFCC. Therefore, the spectrogram possesses more information for the CNN to explore, if training is successful. This result confirms the common conjecture that lower level features usually are preferable as the CNN inputs.

Finally, when comparing the accuracy between SCNN and CLSTM, the SCNN is better although both use convolutional layers. Because the SCNN structure has more layers, the simulation results, in a sense, indicate that the network architecture is more important than the processing power of each node.

To check if the order of accuracy among the models remains the same for other datasets, we repeat the above experiment using the FMA dataset mentioned previously. The experimental results are shown in Table 3. Because the ECNN has a low accuracy, this model is not tested with the FMA dataset. When comparing the accuracy order of the models in Tables 2 and 3, we know the order almost remains the same except MCNN and SLSTM, although the actual accuracy of all models is relatively lower. Because all audio clips in this dataset are all from different soundtracks, such an arrangement avoids the possibility of having two samples in the dataset to have strong correlation. In this regard, we know that the FMA dataset is more difficult than the Jamendo dataset in terms of vocal detection.

Table 3 Accuracy for various network models trained and tested using FMA dataset

Network model	Average accuracy (10 trials) (%)
MCNN	83.7
SCNN	88.2
SLSTM	84.3
CLSTM	85.8
Capsule net	74.9

Table 4 Results of misclassification for various network models using FMA dataset

	Misclassify vocal as non-vocal	Misclassify non-vocal as vocal
MCNN	463.4	211.4
SCNN	238.6	249.4
SLSTM	304.8	347.8
CLSTM	361.4	227.9
Capsule net	579.3	462.3

Table 5 Accuracy for interchanging training and testing dataset

	Training: Jamndo Testing: FMA (%)	Training: FMA Testing: Jamendo (%)
MCNN	76.48	85.72
SCNN	81.69	89.62
Capsule net	70.37	73.36

Since our datasets do not have equal number of vocal and non-vocal segments, it is worthwhile to investigate whether the class imbalance affects the detection accuracy. To this end, we also report the number of mis-classified segments for various network models to detect the FMA dataset in Table 4. Again, the average numbers over 10 trials are given in the table. Recall that the FMA training set has 7451 non-vocal segments and only 4783 vocal segments. However, the results show that there is not a specific type of error dominating the overall error segments in most of the studied networks. Therefore, it seems safe to claim that most network models are not sensitive to the problem of having unequal number of vocal and non-vocal segments.

It is also useful to understand if different datasets affect the generalization capability of the networks. To do so, the network models are trained in one dataset and tested with another dataset. The results are given in Table 5. To save space, only the results of three models are reported. It can be seen that a network trained with Jamendo dataset does not give a very good result for the vocal detection of FMA dataset. On the contrary, a network trained with FMA dataset performs better when tested with Jamendo dataset. In a sense, it implies that the FMA training set has better generalization capability than the Jamendo training set. Therefore, the FMA dataset is a better dataset for vocal detection experiments.

Simulation results for ensemble learning

Experimental results for voting

To test the detection accuracy using voting, we perform the following experiment with two settings. One with all five models involved in voting, whereas the other one with only three models with higher accuracy (namely, SCNN, MCNN, and CLSTM). The experimental results are shown in Table 4. During this experiment, the weights of the models are obtained directly from the previous experiment. As the weights initialization introduces accuracy fluctuations, a fair comparison would be to reuse previously trained weights. When comparing the accuracy of individual model with voting accuracy, we notice that voting can effectively improve the accuracy by 2.4% for Jamendo dataset and 0.7% for FMA dataset, respectively. It is also interesting to know that the accuracy improvement with voting is related to the dataset. The accuracy of FMA dataset appears to be more difficult to improve even with the voting type of stacked ensembles.

Experimental results for post classifier

We also follow similar experimental steps to conduct experiments for the post classifier method. To train the post classifier, we tried two approaches. Approach one is to reuse the training dataset to train the post classifier. For this approach, the advantage is that no additional new training samples are needed to train the post classifier. However, the downside is that the trained classifiers are extremely accurate when classifying the previously seen training samples, typically around 99%. Thus, rendering the post classifier little room to judge which classifier is more reliable in a certain type of data when making the final decision.

With this consideration, we also tried an alternative method. In this alternative approach, we use only 90% of the training samples to train each individual model, and leave the rest 10% for training the post classifier. The experimental results are shown in Table 6. The results reveal that leaving 10% training samples for the post classifier does not always yield better accuracy. In addition, the post classifier method does not outperform the voting approach for all datasets, and thus is not computationally cost-effective.

Table 6 Accuracy for voting and post-classifier types of stacked ensembles

Stacking method	Jamendo dataset	FMA dataset (%)
Voting, 5 models	94.2%	88.9
Voting, 3 models	93.8	88.6
Post classifier, 5 model, approach 1	94.3%	88.7
Post classifier, 3 model, approach 1	93.8%	88.9
Post classifier, 5 model, approach 2	94.3%	89.1
Post classifier, 3 model, approach 2	93.0%	88.5

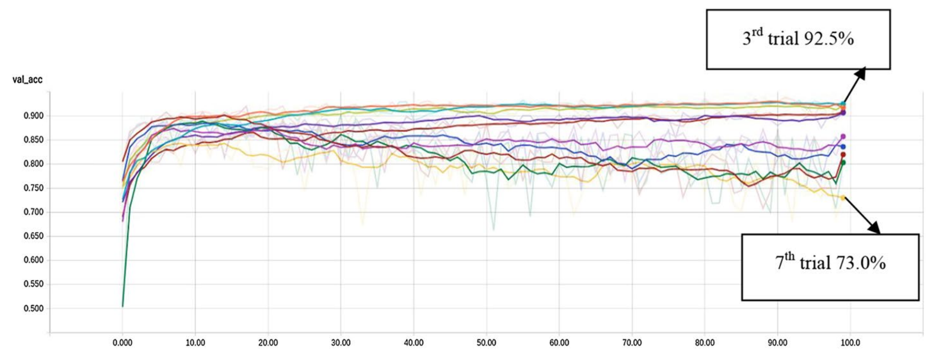

Fig. 14 Accuracy of fusion for various trials

Experimental results for fusion

Another type of stacked ensembles studied in this paper is fusion. In this approach, because the models need different types of feature planes, we must be careful not to shuffle the sequence of the audio clips when computing the feature planes. When trying to train this mega network, despite spending very long period of training time, we find that sometimes the training fails. As shown in Fig. 14, some training trials yield high accuracy after 100 epochs, such as trial 7 with accuracy of 92.4%. However, some other trials have very low accuracy, even down to 73%. When closely examining the loss function, we find that the trials with low accuracy are accompanied with early (after only a couple epochs) loss function increase. It is a signal of training fail. We tried many methods to overcome this problem, such as reducing the learning rate, but none is effective to solve this problem. Eventually, the training time becomes very long, and we are unable to afford such a long waiting for training. In terms of datasets, we tried both Jamendo and FMA datasets. Both have similar tendency to fail in training. For the successful training trails, unfortunately, we are not able to observe any accuracy advantage over the simple voting algorithm either. Therefore, it seems safe to claim that the fusion approach is not (computationally) cost-effective at all when compared with the simple voting approach.

Conclusions

In this paper, we study six different models, namely MCNN, SCNN, ECNN, SLSTM, CLSTM, and capsule net, for singing voice detection. In addition, we also study voting, post classifier, and fusion types of stacked ensembles. The simulation results show that the end-to-end approach (ECNN) has lower accuracy than other models presented in this paper, possibly due to insufficient training samples. Among the network models, the SCNN yields the best accuracy. In addition, a simple voting based on the decisions of five models can increase the accuracy up to 2.4% for the Jamendo dataset. In conclusion, if the accuracy is the top concern, then using multiple network structures with voting is a promising method. In the present study, networks involved in voting are heterogeneous. In the future, we plan to study whether higher accuracy could be achieved by voting with multiple identical network structures, where each structure is trained with either different frequency resolution or a subset of the entire training dataset.

Authors' contributions

All of the authors supervised the experiments and analyzed the results. SDY initialized the study and write the draft version of the manuscript, and the rest two revised the manuscript. All authors read and approved the final manuscript.

Acknowledgements

The authors would like to thank Mr. Chih-Chun Liu and Mr. Ren-Jie Liu for conducting the experiments.

A portion of the materials in this paper was presented in the 7th IEEE International Symposium on Next-Generation Electronics (ISNE 2018), Taipei, Taiwan, May 7–9, 2018 [27].

Competing interests

The authors declare that they have no competing interests.

Availability of data and materials

Not applicable.

Funding

The research was supported in part by the Ministry of Science and Technology (MOST) of Taiwan through Grants MOST 106-2221-E-027-127 and MOST 107-2221-E-027-100.

Publisher's Note

Received: 5 September 2018 Accepted: 8 November 2018
Published online: 26 November 2018

References

1. You SD, Wu Y-C, Peng S-H (2016) Comparative study of singing voice detection methods. Multimedia Tools Appl 75(23):15509–15524
2. Hsu C-L, Wang D, Jang JSR, Hu K (2012) A tandem algorithm for singing pitch extraction and voice separation from music accompaniment. IEEE Trans Audio Speech Lang Process 20(5):1482–1491
3. Logan B, Chu S (2000) Music summarization using key phrases. In: Proceedings of IEEE international conference on acoustics, speech, and signal processing, 2000
4. Salamon J, Gómez E, Ellis DP, Richard G (2014) Melody extraction from polyphonic music signals: approaches, applications, and challenges. IEEE Signal Process Mag 31(2):118–134
5. Kim Y E, Whitman B (2002) Singer identification in popular music recordings using voice coding features. In: Proceedings of the 3rd international conference on music information retrieval, 2002
6. Berenzweig AL, Ellis DP (2001) Locating singing voice segments within music signals. In: IEEE workshop on the applications of signal processing to audio and acoustics, 2001
7. Lukashevich H, Gruhne M, Dittmar C (2007) Effective singing voice detection in popular music using arma filtering. In Workshop on Digital Audio Effects (DAFx'07), 2007
8. Song Y, Kim I (2018) DeepAct: a deep neural network model for activity detection in untrimmed videos. J Inform Process Syst 14(1):150–161. https://doi.org/10.3745/JIPS.04.0059
9. Yu N, Yu Z, Gu F, Li T, Tian X, Pan Y (2017) Deep learning in genomic and medical image data analysis: challenges and approaches. J Inform Process Syst 13(2):204–214. https://doi.org/10.3745/JIPS.04.0029
10. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, 2012
11. Koo KM, Cha EY (2017) Image recognition performance enhancements using image normalization. Human-centric Comput Inform Sci 7(1):33
12. Davis SB, Mermelstein P (1980) Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. IEEE Trans Acoust Speech Signal Process 28(4):357–366
13. Dieleman S, Schrauwen B (2014) End-to-end learning for music audio. In: IEEE international conference on acoustics, speech and signal processing, 2014
14. Dai J, Liang S, Xue W, Ni C, Liu W (2016) Long short-term memory recurrent neural network based segment features for music genre classification. In: 10th International Symposium on Chinese Spoken Language Processing (ISCSLP), 2016
15. Xingjian S H I, Chen Z, Wang H, Yeung D Y, Wong W K, Woo W C (2015) Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In: Advances in neural information processing systems, 2015
16. Sabour S, Frosst N, Hinton GE (2017) Dynamic routing between capsules. In: Advances in neural information processing systems, 2017
17. Kim H G, Sikora T (2004) Comparison of MPEG-7 audio spectrum projection features and MFCC applied to speaker recognition, sound classification and audio segmentation. In: IEEE international conference on acoustics, speech, and signal processing, 2004
18. Rocamora M, Herrera P (2007) Comparing audio descriptors for singing voice detection in music audio files. In: 11th Brazilian symposium on computer music, San Pablo, Brazil, 2007
19. Dittmar C, Lehner B, Prätzlich T, Müller M, Widmer G (2015) Cross-version singing voice detection in classical opera recordings. In: International society for music information retrieval conference (ISMIR), Malaga, Spain, 2015

20. Vembu S, Baumann S (2005) Separation of vocals from polyphonic audio recordings. In: 6th international conference on music information retrieval (ISMIR 2005), London, 2005
21. Nwe T L, Shenoy A, Wang Y (2004) Singing voice detection in popular music. In: Proceedings of the 12th annual ACM international conference on Multimedia, 2004
22. Leglaive S, Hennequin R, Badeau R (2015) Singing voice detection with deep recurrent neural networks. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2015
23. Schlüter J, Grill T (2015) Exploring Data Augmentation for Improved Singing Voice Detection with Neural Networks. In: Proc. of the 16th International Society for Music Information Retrieval Conference, 2015
24. Humphrey E J, Bello J P, LeCun Y (2012) Moving Beyond Feature Design: Deep Architectures and Automatic Feature Learning in Music Informatics. In: Proceedings of the 13th International Society for Music Information Retrieval Conference, Porto, Portugal, 2012
25. Lee J, Park J, Kim KL, Nam J (2018) Samplecnn: end-to-end deep convolutional neural networks using very small filters for music classification. Appl Sci 8(1):150
26. Lim M, Lee D, Park H, Kang Y, Oh J, Park JS, Kim JH (2018) Convolutional neural network based audio event classification. KSII Trans Internet Inform Syst 12(6):2748–2760
27. Huang HM, Chen WK, Liu CH, You SD (2018) Singing voice detection based on convolutional neural networks. In: 2018 7th international symposium on next generation electronics, Taipei, 2018
28. Wu Y C, Chang P C, Wang C Y, Wang J C (2017) A symmetrie kernel convolutional neural network for acoustic scenes classification. In: IEEE international symposium on consumer electronics, Kuala Lumpur, Malaysia, 2017
29. Available https://en.wikipedia.org/wiki/Spectrogram. Accessed 6 Sep 2018
30. Ramona M, Richard G, David B (2008) Vocal detection in music with support vector machines. In: IEEE international conference on acoustics, speech and signal processing, 2008
31. Defferrard M, Benzi K, Vandergheynst P, Bresson X (2017) FMA: a dataset for music analysis. In: 18th international society for music information retrieval conference, 2017
32. Available: https://github.com/NTUT-LabASPL/FMA-C-DataSet-for-Vocal-Detection. Accessed 18 Oct 2018
33. Available: https://www.tensorflow.org/. Accessed 6 Sept 2018
34. Available: https://keras.io/. Accessed 6 Sept 2018
35. Zeiler MD (2012) ADADELTA: an adaptive learning rate method. In; arXiv preprint arXiv:1212.5701
36. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958