Singing Voice Detection using VGGish Embeddings

Shayenne Moura, Marcelo Queiroz
Universidade de São Paulo, Brasil

1 Introduction

The singing voice is one of the principal elements in western music[8]. It carries the lyrics, mood and artist's timbre. Singing voice detection has applications in various MIR tasks such as automatic singer recognition[1], lyrics alignment[6] and melody extraction[4].

In this work we are exploiting a deep learning model (VGGish) trained for a different task (for which there is a lot of labeled data available), in order to compute embeddings that we use as features to train a classic model on a task for which we have much less labeled data available. VGGish [3] features have recently become a popular audio embedding in MIR literature (see[5]), and to the best of our knowledge, we are the first to apply VGGish embeddings on this task.

2 Method

Our goal in this work is to compare VGGish embeddings against standard MFCC features for singing voice detection. We selected a dataset containing singing voice, split it into train/validation/test subsets and trained models to classify audio segments as singing/non-singing.

Audio features are calculated in 0.96 second segments, with 0.48 seconds overlap. We choose this segment size to make possible the use of VGGish default parameters.

Regarding MFCC, we calculate 40 coefficients and used the first 13 (excluding the 0th coefficient). We calculate MFCC features using 10 ms windows and summarize every 96 frames (96*10ms) using the following summary statistics: mean, standard deviation and median, and including delta and double delta, in order to preserve temporal context (feature dimensionality is 13 * n_statistics). The MFCC features are calculated using Librosa [7] 0.5.1.

The ground-truth was based on instrument activations, as defined in the MedleyDB dataset [2]. We consider that a 960 ms segment has singing voice if at least 50% of it has active voice, not necessarily continuous. The types of singing voice included in our dataset are: female singer, male singer, vocalist and choir sources.

We conduct the error analysis over the MedleyDB dataset [2]. We evaluate two classification models using either MFCC or VGGish features: a support vector machine (SVM) and a random forest (RF) classifier. For the SVM we experiment with the C hyper-parameter and choose the optimal value based on the validation set. We do the same for the number of estimators in the RF. Finally, the models are evaluated against the test set using the best model hyper-parameters identified using the validation set.

We evaluate the models quantitatively, verifying the classification accuracy, and qualitatively, comparing the false positives and negatives, listening to the files with low accuracy. Finally, we discuss the results to gain some intuition on the models' performance.

3 Evaluation

3.1 Dataset

The experiments are based on the MedleyDB [2] dataset, using tracks containing singing voice.

We selected the 61 tracks containing singing voice and split them into train, validation and test subsets. The split was made as follow: 80% for train subset and 20% for test subset. Then, we got the train subset and split into 80% for training and 20% for validation. We have 38, 10, 13 songs, for train, validation and test subsets, respectively. This results in 15221, 6147, and 3874 segments (each with a duration of 960 ms) for train, validation and test, respectively.

To avoid the artist/album effect in our classification experiments [9], we used the medleydb API[1] to make the split with artist conditional division, i.e. the subsets do not share the same artist.

3.2 Results

We use the classification accuracy metric to evaluate the performance of the trained models. Table 1 presents the results using different C-values on SVM training phase, evaluated on the validation set. Table 2 shows the results using different number of estimators on RF training phase, evaluated on validation set.

C value	MFCC	VGGish
10	0.84	0.88
1	0.85	0.89
0.1	0.85	0.89
0.01	0.79	0.87

Table 1: C value in SVM to validation set

Estimators	MFCC	VGGish
100	0.83	0.88
200	0.84	0.88
500	0.84	0.88
1000	0.83	0.89

Table 2: No. of estim. in RF to validation set

We choose the models with best results based on the classification accuracy over the validation subset: SVM with C value equals to 0.1 and RF with 500 estimators for MFCC and 1000 estimators for VGGish. We evaluate them on the test set. Table 3 shows the accuracy values.

Our baseline for comparison is the zero-rule classifier that always predicts the majority class, in our case always predicting that singing voice is present.

Classifier	MFCC	VGGish
Baseline	0.71	0.71
Best SVM	0.76	0.83
Best RF	0.75	0.83

Table 3: Bests results models applied to test subset

We see that for both classification models, using VGGish over MFCC features improves classification accuracy by about 8 points. When using VGGish embeddings there is no difference in performance between the SVM and the RF, both obtaining an accuracy of 0.83.

After a qualitative analysis, we notice that specific sound sources (e.g. synthesizer) are classified as singing voice using VGGish and MFCC features. We believe that some sound sources are frequently present when singing voice is active, and the models associate these sources with the desired objective. In the future we could train the models with different mixes of songs could improve the classification, removing spurious source modeling.

4 Conclusions

In this abstract we describe the initial results of our experiments where we evaluate the use of VGGish embeddings to perform singing voice detection. We used SVM and Random Forest models to classify the singing voice segments from MedleyDB and compare the results to using a standard feature (MFCC). Our results show that VGGish embeddings increases classification accuracy by about 8 point in comparison with MFCC to perform singing voice detection, for SVM and Random Forest models on test set. For future directions, we plan to augment the dataset and test if VGGish embeddings and MFCC features are complementary, combining VGGish with other audio features and using cross-validation for evaluation.[2]

[1] `https://github.com/marl/medleydb`

[2] References: link to references file here.