Knowledge Distillation for Singing Voice Detection

Soumava Paul, Gurunath Reddy M, K Sreenivasa Rao and Partha Pratim Das

Indian Institute of Technology, Kharagpur

soumavap@iitkgp.ac.in, {mgurunathreddy, ksrao}@sit.iitkgp.ernet.in, ppd@cse.iitkgp.ac.in

Abstract

Singing Voice Detection (SVD) has been an active area of research in music information retrieval (MIR). Currently, two deep neural network-based methods, one based on CNN and the other on RNN, exist in literature that learn optimized features for the voice detection (VD) task and achieve state-of-the-art performance on common datasets. Both these models have a huge number of parameters (1.4M for CNN and 65.7K for RNN) and hence not suitable for deployment on devices like smartphones or embedded sensors with limited capacity in terms of memory and computation power. The most popular method to address this issue is known as knowledge distillation in deep learning literature (in addition to model compression) where a large pre-trained network known as the teacher is used to train a smaller student network. Given the wide applications of SVD in music information retrieval, to the best of our knowledge, model compression for practical deployment has not yet been explored. In this paper, efforts have been made to investigate this issue using both conventional as well as ensemble knowledge distillation techniques.

Index Terms: Singing Voice Detection, Accelerating CNNs/RNNs, Knowledge Distillation, Soft Labels, Ensemble Knowledge Distillation

1. Introduction & Related Work

Singing voice detection is a binary classification problem in music information retrieval, where the task is to identify singing voice in an audio segment of duration 100 to 200 ms. Efficient voice detection systems can also aid in other MIR tasks such as melody extraction [1, 2] or artist recognition [3]. Early voice detection approaches [4, 5, 6] usually relied on complex hand-engineered audio features, which have now gone out of favour with the advent of deep neural networks. Some examples of such features include MFCCs (Mel-Frequency Cepstral Coefficients), PLPs (Perceptual Linear Predictive Coefficients) and LFPCs (Log Frequency Power Coefficients). These were used as inputs to classification systems like Support Vector Machines, Hidden Markov Models, Random Forests or Artificial Neural Networks. Two of the most popular and recent DNN based approaches [7] and [8] showed that neural networks have the capacity to learn complex features relevant to a particular task such as SVD, from very low-level audio representations such as STFT or Melspectrograms. [9] performed evaluation of these two approaches under a common benchmark protocol so that pre-processing procedures do not give any of the two models an undue advantage in terms of performance.

While these models have considerably improved voice detection performance as well as eliminated reliance on hand-crafted features, little attention has been paid to the optimal structure of these models - whether we actually need the huge memory and computation power needed to train these models. Also, from a deployment point of view, these networks are all non-optimal owing to their sufficiently large inference times. In this paper, we try to address this issue using principles of knowledge distillation [10]. Our experiments show that the optimal CNN-based model is actually more than 250 times smaller than the model used by [8], and the optimal RNN-based model is at least 2.5 times smaller than the one used by [7]. Moreover, we show that, with knowledge distillation on smaller student models, it is possible to obtain accuracies substantially higher than the current state-of-the-art.

2. Proposed Methods

2.1. Basics of Knowledge Distillation

The basic idea behind knowledge distillation is to have the student network trained with not only the information provided by true labels (also called hard targets) but also by using soft targets produced by the teacher network (also referred to as cumbersome model) as a regularizer. This helps the student network learn to mimic the teacher's behaviour. In accordance with [10], a scaling hyperparameter referred to as temperature is used to regulate the softness of the targets from the teacher network.

The generalized formula for softmax is given by:

$$p_i = \frac{\exp\left(s_i/\tau\right)}{\sum_j \exp\left(s_j/\tau\right)} \tag{1}$$

where s_i and p_i are the logit produced for the ith class and the corresponding class probability respectively and τ is the softness regulating temperature.

A KL-divergence loss is taken between the softmax probabilities of the teacher and the student, represented as q and p respectively, raised to temperature τ (between 2-20 in our experiments) and is given by:

$$L_{KD} = \tau^2 KLD\left(q, p\right) \tag{2}$$

The combined loss is given by:.

$$L_{\text{Total}} = (1 - \lambda)L_{CE} + \lambda L_{KD} \tag{3}$$

where L_{CE} is the cross-entropy loss between the student's softmax probabilities (at $\tau = 1$) and the correct labels (hard targets) and λ is a second hyperparameter controlling the trade-off between the 2 losses. Generally, better results are obtained by keeping λ close to 1, i.e., weight on L_{KD} higher.

2.2. Knowledge Distillation with Schluter CNN

Before starting the discussion specific to knowledge distillation, we first describe the model architecture and input features in Schluter CNN (Teacher model) [8]:

2.2.1. Input Features

[8] uses melspectrograms (80-D) of 115 consecutive frames (1.6 seconds) of an audio signal as input feature. The label of the central frame is considered as the label of a particular sample. The resulting input dimension is 80x115. The Mel bank is normalized to have zero mean and unit variance over the training data.

2.2.2. Model Architecture

The Schluter CNN employs three types of feedforward neural network layers: 2D convolutional layers with 3x3 kernels and Leaky ReLU activations [11], maxpooling layers with 3x3 kernels and dense layers. To describe the architecture, we use the following shorthand notations: 'ConvX' denotes a conv layer with X channels and Leaky ReLU activation, 'Max' stands for maxpool, and 'DenseX' denotes a dense layer with X neurons. The architecture can then be described as Conv64-Conv32-Max-Conv128-Conv64-Max-Dense256-Dense64-Dense2. Additionally, there are two dropout layers with dropout probability 0.2 in between the final 3 dense layers. Note that the original model was trained with a sigmoid output activation function and binary cross entropy loss where appropriate thresholds were used on the output probability (between 0 and 1) to infer the presence or absence of a voice label. However, since the main objective of this paper is knowledge distillation, we needed a softmax activation on the output. Hence, in our modified model, the output dimension is 2 with a softmax activation and the model is trained with categorical cross-entropy loss. In Table 2, we compare the performance of our base model with the benchmarks by [9] and show that the two sets of accuracies are almost identical. This justifies the above mentioned modification to Schluter CNN.

2.2.3. Building Student Networks from Base Model

The base model (Schluter CNN) has a total of 14,08,290 ($\sim$ 1.4M) parameters. To build student models, we focus on reducing the number of channels per layer (except maxpool of course). The lowest number of channels in any particular layer in the architecture mentioned above is 32. We define a variable "filter scale" (abbreviated as FS) that can have any of the values in {2, 4, 8, 16, 32}. As the name suggests, we divide the number of channels in each layer by FS to reduce the overall number of model parameters. Higher the value of FS, lower the number of parameters. We denote each resulting student network as FSX. Student networks trained with distillation are denoted as KD-FSX. These notations are later used in Section 3.3. As shown in Table 1, the highest reduction achieved with student networks in terms of parameters is nearly 1000x with FS32. The process of knowledge distillation from teacher CNN model to student CNN model is shown in Fig. 1. As shown in figure, teacher model is the pre-trained Schluter CNN. During distillation, teacher model parameters are frozen and the knowledge of the teacher model is transferred to a small student CNN model with reduced parameters.

Table 1: *Number of parameters in Student Models*

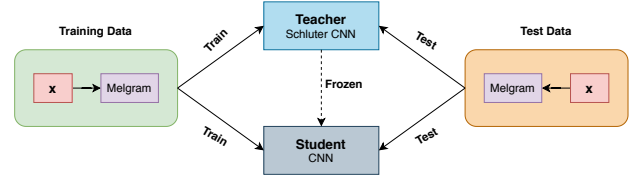| Model	Parameters
FS2	3,52,402
FS4	88,266
FS8	22,150
FS16	5,580
FS32	1417

Figure 1: *Distilling Schluter CNN to student CNN*
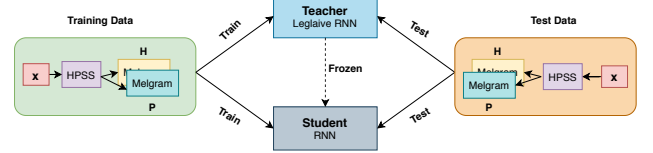

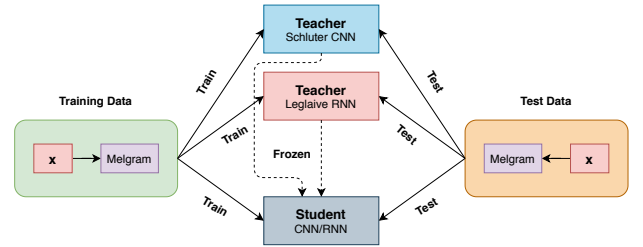Figure 2: *Distilling Leglaive RNN to student RNN*

Figure 3: *Ensemble distillation of Schluter CNN and Leglaive RNN to student CNN and RNN model*

2.3. Knowledge Distillation with Leglaive RNN

Similar to the CNN case, we first describe the model architecture and feature inputs in Leglaive RNN (Teacher model) [7]:

2.3.1. Input Features

Leglaive RNN [7] first applies a double stage harmonic-percussion source separation (HPSS) [12] on the audio signal as pre-processing to extract signals relevant to the singing voice. The main idea behind HPSS is to decompose the spectrogram of the input signal into one spectrogram smooth in time direction (associated to harmonic components), and another spectrogram smooth in frequency direction (associated to percussive components). Features are then extracted from a filter bank on a Mel scale and the resulting melspectrograms of the harmonic and percussive components are concatenated as input to the Bidirectional LSTM (BiLSTM) network. This gives 80-dimensional feature vectors for 218 frames (3.5 seconds) per sample. Additionally and similar to Schluter CNN, the mel bands are normalized to have zero mean and unit standard deviation over the train set. However, unlike Schluter CNN, Leglaive RNN infers the label of each of the 218 time frames.

2.3.2. Model Architecture

The model is made up of 3 layers of stacked BiLSTM units of sizes 30, 20, and 40. Finally, this is followed by a shared dense layer (also known as time distributed dense) to yield predictions for the 218 output time frames (3.5 seconds). Similar to 2.2.2, the original model was trained with a sigmoid output (output dimension = 1) for each of the 218 frames and binary

cross entropy loss. Keeping in mind our main objective of experimentation with distilled student models, we re-implement Leglaive RNN with 2 output dimensions with softmax activation, for each frame. Comparison with [9]'s implementation is shown in Table 5. Almost identical numbers justify these modifications to the teacher network.

2.3.3. Building Student Network from Base Model

Leglaive RNN has a total of 65,682 ($\sim$ 65.7K) parameters. Reducing only the number of LSTM units would not have too much of an effect in terms of parameter count reduction. So, we completely remove the last 2 BiLSTM layers and that gives close to 2.5x reduction in size for the student model with a total of 26,762 ($\sim$ 26.8K) parameters. For future reference, the teacher is abbreviated as LRNN (Leglaive RNN) and the student as SRNN (small RNN). Student network trained with distillation is denoted as KD-SRNN. Knowledge distillation from teacher RNN to student RNN model is shown in Fig. 2. As shown in figure, teacher model is the pre-trained Leglaive RNN. During distillation, teacher model parameters are frozen and the knowledge of the teacher model is transferred to a small student RNN model with reduced parameters.

2.4. Ensemble Knowledge Distillation

In this section, we propose to train both CNN and RNN based student networks using the two teachers Schluter CNN and Leglaive RNN. For this, we use the input features representation used for SVD in [8] without any data augmentation. However, for this feature representation, we do not have any prior RNN based models in literature. To get around this problem, we train the Leglaive RNN architecture on this data, the only difference being, the RNN now predicts labels of the central time frame as in the case of Schluter CNN. The parameter count remains unchanged from 2.3.3. We first benchmark the performance of LRNN and SRNN on the new data in Table 7. The input dimension of Schluter CNN is same as before (80x115); that of Leglaive RNN is 115x80 (time dimension comes first in RNNs). Similar to the proposition in [10], we combine the predictions of the teachers in the ensemble (here 2) by taking the arithmetic or geometric mean of their predictions as soft targets. The CNN based students are denoted as ENKD-FSX and the RNN based one as ENKD-SRNN. These notations are reused in Section 3.6. As shown in Table 8, we achieve state-of-the-art performance on the Jamendo dataset with this formulation. The ensemble knowledge from two teacher models to a student model is shown in Fig. 3. As shown in figure, during knowledge distillation, the teacher (Schluter CNN and Leglaive RNN) model parameters are frozen and the knowledge of both teacher models are transferred to student CNN and RNN models.

3. Experiments

3.1. Dataset and Protocol

For our experiments, we use the Jamendo dataset, a publicly available dataset with singing voice annotations, that was also used by [7] and [8]. It contains 93 copyright-free songs, collected and annotated by [6]. For comparison to existing results [9], we follow the official split of 61 files for training and 16 files each for validation and testing. We use the validation set for hyperparameter tuning and report test results corresponding to the highest validation accuracy. In accordance with [9], for test set, we report accuracy, precision, recall, F-measure, False Positive Rate (FPR) and False Negative Rate (FNR). Our source

code is built on top of a public repository by [9] available here - https://github.com/kyungyunlee/ismir2018-revisiting-svd.

3.2. Reproducing Results of Schluter CNN

Here we compare the results of our implementation (output dimension 2 and trained by categorical cross-entropy loss) with the implementation by [9]. Note that, similar to [9], we also do not apply data augmentation for a fair comparison with [7]. As shown in Table 2, our numbers only differ by small magnitudes.

Table 2: *Comparison of our Schluter CNN implementation with [9]'s implementation*

Model	Acc.	Prec.	Recall	F-Measure	FPR	FNR
[9]	**86.8**	**83.7**	89.1	**86.3**	**15.1**	10.9
Ours	85.4	81.3	**89.3**	85.1	17.9	**10.7**

Table 3: *Performance of Student Derivatives from Schluter CNN*

Model	Acc.	Prec.	Recall	F-Measure	FPR	FNR
FS2	**87.0**	81.8	**92.8**	**86.9**	18.0	**7.2**
FS4	85.2	80.4	90.1	85.0	19.1	9.9
FS8	**87.0**	84.0	89.1	86.5	14.8	10.9
FS16	86.5	84.5	86.6	85.7	13.5	13.4
FS32	83.7	**85.8**	77.9	81.7	**11.2**	22.1

Table 4: *Performance of Student Models with Knowledge Distillation*

Model	Acc.	Prec.	Recall	F-Measure	FPR	FNR
KD-FS2	87.0	83.7	89.4	86.4	15.2	10.6
KD-FS4	**87.3**	**84.5**	88.3	**86.6**	**13.6**	11.7
KD-FS8	86.6	83.1	89.5	86.2	15.9	10.5
KD-FS16	85.0	80.0	**90.4**	84.9	19.7	**9.6**
KD-FS32	81.5	79.4	81.3	80.3	18.4	18.7

Table 5: *Comparison of our Leglaive RNN implementation with [9]*

Model	Acc.	Prec.	Recall	F-Measure	FPR	FNR
[9]	87.5	**86.1**	87.2	86.6	**12.2**	12.8
LRNN (**Ours**)	88.2	85.7	89.7	**87.8**	13.2	**10.3**

Table 6: *Performance Comparison of SRNN and KD-SRNN*

Model	Acc.	Prec.	Recall	F-Measure	FPR	FNR
SRNN	87.6	84.5	90.6	87.5	14.6	9.4
KD-SRNN	**88.9**	**85.7**	**91.5**	**88.5**	**13.4**	**8.5**

Table 7: *Performance of LRNN and SRNN on data used by [8]*

Model	Acc.	Precision	Recall	F-Measure	FPR	FNR
LRNN	**82.3**	**78.2**	**85.8**	**81.8**	20.9	**14.2**
SRNN	76.1	70.5	83.8	76.6	30.6	16.2

3.3. Knowledge Distillation with Schluter CNN

3.3.1. Performance of Student Networks without Distillation

Here we show the performance of the student networks from Table 1. These are trained similar to the teacher network. As shown in Table 3, with our smaller student networks, we get better results across all performance measures, with respect to both ours and the original implementation in Table 2. Interestingly enough, FS32, having 1000x lesser parameters than the teacher, succeeds in improving on 2 performance measures,

Table 8: *Performance Comparison of ENKD Variants*

Model	Acc.		Prec.		Recall		F-Measure		FPR		FNR	
	AM	GM	AM	GM	AM	GM	AM	GM	AM	GM	AM	GM
ENKD-FS2	87.9	87.7	86.3	86.5	87.9	87.1	87.1	86.8	12.1	11.8	12.1	12.9
ENKD-FS4	86.8	**88.4**	84.4	**86.8**	88.0	88.4	86.1	**87.6**	14.2	**11.7**	12.0	11.6
ENKD-FS8	87.7	87.1	86.2	81.9	87.6	**92.8**	86.9	87.0	12.2	17.8	12.4	**7.2**
ENKD-FS16	86.4	86.5	84.2	83.4	87.1	88.1	85.6	85.8	14.3	15.0	12.9	11.9
ENKD-FS32	84.3	84.2	82.3	82.8	84.4	83.3	83.3	83.1	15.8	15.0	15.6	16.7
ENKD-SRNN	84.4	83.1	81.1	81.5	86.5	82.5	83.7	82.0	17.5	16.3	13.5	17.5

Precision and FPR, over Schluter CNN. This clearly indicates, that smaller models can be equally effective in SVD and require further attention.

3.3.2. Performance of Distilled Student Networks

Here we present the performance of the student networks from 3.3.1, trained with distillation by the teacher network. As shown in Table 4, performance falls on almost every measure with the student models. This is expected since, as per Table 3, student networks already have a lesser tendency to overfit, i.e., they actually perform better than Schluter CNN. So, additional supervision from the teacher actually ends up hurting the student network. However, as we show in Section 3.6, performance of all FSX models improve with additional supervision from an LSTM-based [13] teacher network.

3.4. Reproducing Results of Leglaive RNN

Similar to Section 3.2, we compare our implementation of Leglaive RNN with that of [9]. Here the output dimension is 2 for the time distributed dense layer and the model is trained by categorical cross-entropy loss. As shown in Table 5, our numbers only differ by small magnitudes.

3.5. Knowledge Distillation with Leglaive RNN

3.5.1. Performance of Student Network with and without Distillation

As shown in Table 6, KD-SRNN comprehensively outperforms SRNN on all performance measures, showing the benefits of knowledge distillation in this context. Although SRNN performs slightly worse than LRNN on some of the measures, KD-SRNN surpasses LRNN (our implementation) on all measures and is inferior to the original implementation [9], only on Precision and FPR.

3.6. Ensemble Knowledge Distillation

3.6.1. Benchmarking LSTM Performance

First we show the performance of LRNN and SRNN on the data used by Schluter CNN in Table 7. We see that its performance is considerably lower than Schluter CNN or any of the FSX derivatives. Despite this, including it as a teacher in the ensemble considerably improves the performance of all ENKD-FSX and ENKD-SRNN models, as shown in Table 8.

3.6.2. ENKD Models

Here we present the performance of all CNN and RNN based student networks trained by Ensemble Knowledge Distillation shown in Table 8. AM (Arithmetic Mean) and GM (Geometric Mean) denote method by which teacher predictions were combined. With Ensemble Knowledge Distillation (ENKD), we

get substantial improvements over all CNN and RNN models trained by simple knowledge distillation (KD). On accuaracy, ENKD improves KD-FSX variants by the following amounts - 0.9% (FS2), 1.1% (FS4), 1.1% (FS8), 1.5% (FS16), 2.8% (FS32). ENKD also gives a new state-of-the-art model on the features used by [8], ENKD-FS4 (mainly based on accuracy) that improves current best reported results (Table 2) by 1.6%. This technique also proves quite beneficial for SRNN, improving its accuracy by 8.3% to 84.4% (higher than even LRNN). However, this accuracy is still lower than that of ENKD-FS4 by a fair amount, proving that CNN is still the better model for this kind of data.

In summary, smaller student networks built from Schluter CNN actually show lesser overfitting tendencies and hence perform better than the base model in several cases. That is why knowledge distillation experiments with the base model as a teacher, fail to get any substantial improvement over the student networks' performance. Knowledge Distillation experiments with Leglaive RNN produce new state-of-the-art results on the features used by [7], with a smaller student model, SRNN. With Ensemble Knowledge Distillation (ENKD), we get substantial improvements over all CNN and RNN models trained by simple knowledge distillation (KD).

4. Conclusion and Future Work

In this paper, we have shown that application of knowledge distillation techniques to compress the singing voice detection models can be a new direction of research in the field of Music Information Retrieval. Our experiments show that smaller models trained with distillation can achieve comparable and in some cases, even higher accuracies than current state-of-the-art models. With respect to knowledge distillation, another interesting experiment can be training student models on a subset of the training data and comparing their performance with student/teacher models that learn from the entire data. We also plan to explore multi-step knowledge distillation techniques in the future, like the Teacher Assistant model [14], where an intermediate-sized network can supposedly help bridge the knowledge gap between student and teacher better and hence aid in more efficient distillation.

5. Acknowledgements

6. References

[1] C.-L. Hsu, L.-Y. Chen, J.-S. R. Jang, and H.-J. Li, "Singing pitch extraction from monaural polyphonic songs by contextual audio modeling and singing harmonic enhancement," in *ISMIR*, 2009.

[2] M. G. Reddy and K. S. Rao, "Predominant melody extraction from vocal polyphonic music signal by combined spectro-

temporal method," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 455–459.

[3] A. Berenzweig, D. P. Ellis, and S. Lawrence, "Using voice segments to improve artist classification of music," *Academicommons*, 2002.

[4] B. Lehner, R. Sonnleitner, and G. Widmer, "Towards light-weight, real-time-capable singing voice detection." in *ISMIR*, 2013, pp. 53–58.

[5] B. Lehner, G. Widmer, and R. Sonnleitner, "On the reduction of false positives in singing voice detection," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 7480–7484.

[6] M. Ramona, G. Richard, and B. David, "Vocal detection in music with support vector machines," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2008, pp. 1885–1888.

[7] S. Leglaive, R. Hennequin, and R. Badeau, "Singing voice detection with deep recurrent neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 121–125.

[8] J. Schlüter and T. Grill, "Exploring data augmentation for improved singing voice detection with neural networks." in *ISMIR*, 2015, pp. 121–126.

[9] K. Lee, K. Choi, and J. Nam, "Revisiting singing voice detection: A quantitative review and the future outlook," *arXiv preprint arXiv:1806.01180*, 2018.

[10] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[11] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.

[12] N. Ono, K. Miyamoto, J. Le Roux, H. Kameoka, and S. Sagayama, "Separation of a monaural audio signal into harmonic/percussive components by complementary diffusion on spectrogram," in *2008 16th European Signal Processing Conference*. IEEE, 2008, pp. 1–4.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] S.-I. Mirzadeh, M. Farajtabar, A. Li, and H. Ghasemzadeh, "Improved knowledge distillation via teacher assistant: Bridging the gap between student and teacher," *arXiv preprint arXiv:1902.03393*, 2019.