SPEAKER VERIFICATION WITH APPLICATION-AWARE BEAMFORMING

Ladislav Mošner, Oldřich Plchot, Johan Rohdin, Lukáš Burget, Jan Černocký

Brno University of Technology, Faculty of Information Technology, IT4I Centre of Excellence, Czechia

ABSTRACT

Multichannel speech processing applications usually employ beamformers as means of speech enhancement through spatial filtering. Beamformers with learnable parameters require training to minimize a loss function that is not necessarily correlated with the final objective. In this paper, we present a framework employing recent neural network based generalized eigenvalue beamformer and application-specific model that allows for optimization of beamformer w.r.t. target application. In our case, the application is speaker verification which utilizes a speaker embedding (x-vector) extractor that conveniently comes with desired loss. We show that application-specific training of the beamformer brings performance improvements over a system trained in the standard way. We perform our analysis on the recently introduced VOiCES corpus which contains multichannel data and allows us to modify the evaluation trials such that enrollment recordings remain single-channel and test utterances are multichannel.

Index Terms— Speaker verification, beamforming, x-vector, generalized eigenvalue problem

1. INTRODUCTION

Performances of speaker recognition (SR) systems have significantly improved in the last three years, mainly due to the introduction of x-vectors [1] which have gradually replaced i-vectors [2] and became a new state-of-the-art technique. X-vectors bring a new and very convenient scheme of discriminative training while the Probabilistic Linear Discriminant Analysis [3] is still kept as a generative backend on top of x-vectors to perform the actual speaker verification task and, in the end, to provide scores for speaker verification trials. Speaker verification with far-field data is still challenging, but x-vectors nowadays dominate also this domain [4]. The reason why far-field SR is complicated is mainly because distortions of the acoustic speech signal are introduced when the propagating sound wave gets reflected on walls and obstacles. Given the different absorption properties of various materials, it is attenuated and then returned to the room, resulting in a reverberated signal. Therefore, when using distant microphones and more specifically arrays of such microphones, we obtain multiple distorted and reverberated copies of the original speech signal that are time-shifted and often barely usable when using each of these channels individually.

Dealing with reverberated and distorted data has motivated the development of various speech enhancement methods [5] that are unaware of the target application, but can be used universally as a pre-processing step. When dealing with microphone arrays and multiple parallel channels, various beamforming techniques [6] are typically used. Usually, these models are trained to optimize the signal to noise ratio (SNR) of their output or the Minimum Variance Distortionless Response (MVDR) criterion [7]. The goal of such methods is to produce an enhanced output for the human listener and the mean opinion score (MOS) or PESQ (Perceptual Evaluation of Speech Quality) are usually used as measures of quality. Even though these beamformers are trained separately to enhance the *perceptual* quality of speech, they have been proven to provide substantial improvement when used as a prepossessing step for the *automatic* speech recognition (ASR) system [8]. In order to make the beamforming aware of the target application, an ASR DNN model was prepended with multichannel time convolution filters that take raw audio signals and then map this multichannel input by means of filter-and-sum down to a single channel that is then fed to the ASR model [9, 10, 11]. This way, the spatial filtering was trained together with the ASR acoustic model by optimizing the cross-entropy objective. This approach, however, assumes a fixed configuration of input channels and also requires large amounts of training data. Recent research in acoustic beamforming that employs neural networks (NN) [12] together with Generalized Eigenvalue (GEV) Beamformer has been introduced in [13] to overcome the problem of having a fixed channel configuration and it has been successfully trained jointly with an ASR system [14].

In this work, we focus on processing of the speech captured with irregularly shaped distant microphone arrays in such a way that the final output is tuned for speaker verification. In order to make the beamformer aware of our target application, we will make use of the GEV beamformer with an NN based spectral mask estimator [13]. We will implement

The work was supported by Google Faculty Research Award program, by Czech Ministry of Interior project No. VI20152020025 "DRAPAK" and by Czech Ministry of Education, Youth and Sports from the National Programme of Sustainability (NPU II) project "IT4Innovations excellence in science - LQ1602".

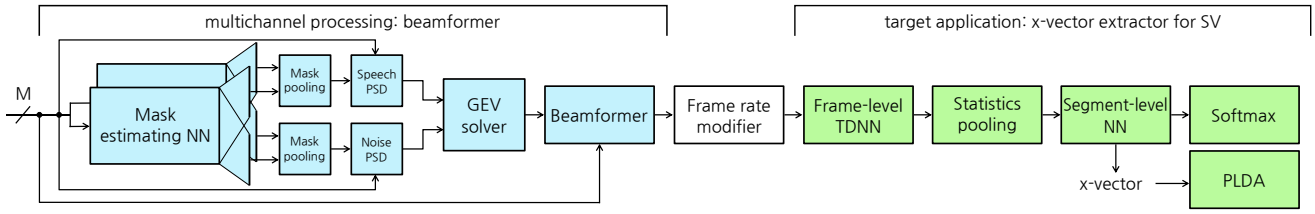

Fig. 1. Proposed framework combining multichannel processing (beamforming) and target application (x-vector extractor for a speaker verification task). Mask estimator is updated using the loss propagated from the target application model.

the whole structure as Tensorflow model and connect it to the state-of-the-art x-vector extractor which is trained to discriminate between a set of training speakers. In our case, we will be unable to train both models jointly as we simply do not have a large-enough database of multi-channel training data. We will use an already existing x-vector model whose parameters will be fixed during training. The exception will be the last layer of the x-vector model, which we allow to be trained for a few iterations on the set of speakers in the multichannel dataset. The beamformer will be either randomly initialized or pre-trained and we will experiment with two strategies of its training: i) optimizing the mean squared error (MSE) between the source and beamformed signal and ii) optimizing the binary cross-entropy (BCE) between the values in estimated spectral masks and precomputed ideal binary masks (IBM) [13]. When using randomly initialized mask estimator parameters, we will optimize only the cross-entropy objective at the end of the x-vector model.

2. METHOD

The framework we present comprises two components as shown in Figure 1. The first one is responsible for multichannel processing to *enhance* speech contained in the input signals. Therefore, the input to the first model consists of multiple signals. They are usually obtained with a microphone array. Nowadays, home devices make use of microphone arrays where the position of microphones is fixed and known in advance and their properties can be theoretically described. We deal with a more general scenario, where our microphone array comprises microphones spread around in a room at unknown locations (even though the positions have been disclosed for our particular evaluation data [15], we take no advantage of this knowledge in this work and perform a blind enhancement).

The second component is a model specific for the target application. It is fed only with a single channel input coming from the enhancer. The second model can be for instance an acoustic model in automatic speech recognition (ASR) or a DNN embedding extractor trained for the purposes of various speech tasks such as language identification (LID) or speaker verification (SV). In this work, we aim at speaker verification hence the downstream model is a neural network mapping a sequence of input features to a single fixed-length vector characterizing the identity of a person speaking in an observed utterance.

A common requirement for both components is that they represent some functions with learnable parameters that are to be updated by means of gradient descent method to optimize a particular objective function.

2.1. Beamformer – speech enhancement component

Generally, beamforming is a method that performs spatial filtering aiming at emphasizing sounds coming from a direction where the speaker of interest is located. At the same time, sounds from other directions are suppressed. We will consider a beamformer operating in a frequency domain. Assuming that the multichannel input is in the short-time Fourier transform (STFT) domain, then $\boldsymbol{y}_{f,t}$ is an M-dimensional vector where M denotes number of input channels (microphones), f indicates the frequency bin, and t is the time index. The beamformer then attempts to recover the original signal or a signal at a reference microphone by performing linear combination of the channels in $\boldsymbol{y}_{f,t}$ via an M-dimensional complex weight vector $\boldsymbol{w}_f$ (we consider time independent weight vector hence the time index is omitted). The output $s_{f,t}$ is then obtained as

$$s_{f,t} = \boldsymbol{w}_f^H \boldsymbol{y}_{f,t}. \tag{1}$$

The superscript $(\cdot)^H$ stands for Hermitian transpose. The weight vector is usually obtained by optimization of a particular criterion such as minimization of a signal variance subject to a *distortionless constraint* — minimum variance distortionless response (MVDR) [7].

In this work, we adopt a generalized eigenvalue beamformer (GEV) [16] that maximizes the signal-to-noise ratio (SNR) objective:

$$\boldsymbol{w}^{GEV} = \arg\max_{\boldsymbol{w}} \frac{\boldsymbol{w}^H \boldsymbol{\Phi}_{XX} \boldsymbol{w}}{\boldsymbol{w}^H \boldsymbol{\Phi}_{NN} \boldsymbol{w}}. \tag{2}$$

$\boldsymbol{\Phi}_{XX}$ denotes Power Spectral Density matrix (PSD) of speech and $\boldsymbol{\Phi}_{NN}$ stands for PSD of additive noise. All terms in (2) are frequency-dependent and for the sake of brevity, from now on, we omit the f subscript. We chose the GEV beamformer for multiple reasons:

- It revealed a great performance for ASR in the 4-th CHiME challenge [8] and moreover, it has been suc-

cessfully combined with an acoustic model and trained jointly [14].

- It satisfies the requirement that the front-end model allows for optimization of its learnable parameters. Heymann showed in [13] an approach to estimation of PSD matrices $\mathbf{\Phi}_{XX}$ and $\mathbf{\Phi}_{NN}$ by means of original spectrum masking where the masks are estimated by a neural network. Therefore, weights of the network constitute learnable parameters of enhancing multichannel front-end.

- The approach with neural network based masks estimation allows performing beamforming with varying number of microphones. This is because networks processing individual channels share weights. This enhances the generality of the framework.

Neural network supported beamformer relies on estimated masks as shown in Figure 1. A mask is estimated for each channel and then they are combined using median or mean operation to obtain the same mask for all the channels which proved to be useful [17]. To avoid sparsity of gradients [14], we employ mean pooling.

The GEV beamformer coined its name by the fact that optimization of (2) leads to a generalized eigenvalue problem

$$\mathbf{\Phi}_{XX}\boldsymbol{w} = \lambda\mathbf{\Phi}_{NN}\boldsymbol{w} \tag{3}$$

with λ denoting eigenvalue and $\boldsymbol{w}$ eigenvector. Therefore, beamformer weights are given by the eigenvector corresponding to the largest eigenvalue. The fact that the whole beamformer must be incorporated in the final model results in a requirement of implementing the gradient propagation through a generalized eigenvalue decomposition.

2.2. X-vector extractor – application-specific component

An x-vector extractor [1] is a DNN architecture for extraction of fixed size speaker representations from utterances of variable durations. It consists of one block that operates on the framewise features, followed by a statistics pooling layer that calculates the mean and standard deviation over its input, followed by a standard DNN that takes the fixed size statistics from the pooling layer as input and predicts the identity of the speaker of the utterance. As in the original work [1], we use time-delay neural network layers (TDNN) for the frame-level processing which makes it possible to use a large context without increasing the number of parameters too much.

2.3. Component connection and gradient propagation

Both models need to be joint in such a way that the back-propagation continues to the first model. In our scenario, we assume that the application-specific model is trained and can back-propagate error to the beamforming component. Only parameters of the mask estimator are updated while keeping x-vector extractor weights fixed. The beamformer outputs frames of a complex spectrum. The x-vector extractor we use requires Mel-frequency Cepstral (MFCC) coefficients as its input. The issue with the downstream model is that its frame rate may differ from that of a beamformer which is our case as well. Even though, according to [14], change of mask estimator frame rate to match frame rate of an acoustic model has no significant impact on speech recognition results, we opted for more general approach to be able to support various downstream models. Therefore, inverse STFT is applied to the beamformer output to obtain time domain signal and transform it back to STFT domain with a different frame size and shift. These operations are differentiable. All these operations are referred to as *frame rate modifier* in Figure 1.

As displayed in Figure 1, the eigenvalue decomposition stands in between the mask estimator and output of the beamformer. In order to update weights of the mask estimator using error propagated from x-vector extractor, it is required to be able to take the derivative of eigenvalue decomposition w.r.t its input. This is not a straight forward task [18]. Neural network frameworks usually support solvers for eigenvalue problem (where the input matrix is Hermitian) but not for the generalized one.

We approach the generalized eigenvalue problem solution by turning it into the eigenvalue problem via Cholesky decomposition of a noise PSD: $\mathbf{\Phi}_{NN} = \boldsymbol{L}\boldsymbol{L}^H$, where $\boldsymbol{L}$ is a lower triangular matrix. The problem defined in (3) is then reformulated as follows:

$$\begin{aligned} \mathbf{\Phi}_{XX}\boldsymbol{w} &= \lambda\boldsymbol{L}\boldsymbol{L}^H\boldsymbol{w} \\ [\boldsymbol{L}^{-1}\mathbf{\Phi}_{XX}(\boldsymbol{L}^H)^{-1}]\boldsymbol{L}^H\boldsymbol{w} &= \lambda\boldsymbol{L}^H\boldsymbol{w} \\ \boldsymbol{A}\boldsymbol{y} &= \lambda\boldsymbol{y} \end{aligned} \tag{4}$$

The last line represents an eigenvalue problem with Hermitian matrix $\boldsymbol{A}$ and thus may be solved using available routines that support gradient propagation. However, there is a need for gradient propagation through Cholesky decomposition we introduced. In Tensorflow[1], that we use for training, it is possible. As follows from (4), beamforming weights are obtained by linear transformation of eigenvector corresponding to the highest eigenvalue: $\boldsymbol{w} := (\boldsymbol{L}^H)^{-1}\boldsymbol{y}$. At this point, the whole generalized eigenvalue problem is differentiable and error from the x-vector extractor can be used to update beamformer parameters.

3. EXPERIMENTAL SETUP

3.1. Data and preparation of multichannel subsets

In both training and testing of the beamformer model, we made use of recently published Voices Obscured in Complex Environmental Settings (VOiCES) corpus [15]. It consists of speech selected from LibriSpeech retransmitted (with rotating source loudspeaker) in multiple rooms and corrupted by

[1] https://www.tensorflow.org

additive replayed noise. In each room, 12 microphones were recording simultaneously.

In order to prepare training, development, and evaluation data, we made use of the definition of development and evaluation sets for the VOiCES challenge [19]. Our training corpus for the mask estimator training is based on a complete set of recordings from room 1 and room 2[2]. From this set, we filtered out the development recordings (as defined by the VOiCES challenge) because they constitute a subset of it. We also removed files where we found an inconsistency in lengths of source LibiSpeech and retransmitted recordings which would cause problems in mask estimator pre-training. In the following step, recordings were grouped to quartets based on speaker identity, chapter, segment, room, and distractor type ensuring that all four microphones recorded the same session. Microphones in microphone arrays were chosen randomly to enhance the diversity of the set. The resulting training dataset consists of 57,800 examples (arrays comprising four microphones) spanning voices of 200 speakers. Average length of utterances is approximately 15 seconds.

Development and evaluation sets are the same as those used in the VOiCES challenge in terms of recordings. Nevertheless, examples in our lists of examples comprise quartets of microphones[3] constituting arrays. As a result, we obtained 3,912 examples containing voices of speakers present in the training set. The original development trials always consist of a clean enrollment utterance and noisy/reverberant test utterance. For every single-channel enrollment recording, 8 test recordings with the same content just recorded with different microphone (over the different channel) exist to form 8 trials. Therefore, the development microphone arrays were created by randomly splitting the 8 channels into two 4-microphone subsets. Hence the multichannel trial set is four times smaller than the original one. Out of 1,001,472 resulting trials, 996,448 are impostor and 5,024 are target.

The VOiCES evaluation set was recorded in distinct acoustic conditions (room 3 and room 4) and contains 100 unique speakers. The process of creation of microphone quartets and trial set resembles the procedure for the development set. The difference is that in the evaluation trial definition, there are always 11 test utterances (differing in channels) per one enrollment utterance. In order to create 4-microphone arrays out of 11 microphones, we randomly selected 2 quartets and remaining 3 recordings were combined with one randomly picked (already used) audio file. Overall, the evaluation set comprises 3,018 test quartets of signals. The difference between the original development and evaluation trials is that part of the enrollment segments is clean and part is noisy/reverberant. We stick with this and do not perform any enhancement of corrupted signals. Out of 983,868 trials, 973,929 are impostor and 9,939 are target.

[2]Data from room 3 and room 4 was not released by the time of our experimentation. The only available data from these rooms was provided for the VOiCES challenge in the form of evaluation set.

[3]We did not experiment with a different number of microphones in training and testing even though it would be possible.

Table 1. Performance of the x-vector extractor that serves as an application-specific model. Prior target probability P_{tar} for the minimun detection cost C_{det} is set to 0.01.

	EER [%]	C_{det}
VOiCES dev	2.03	0.261
VOiCES eval	5.51	0.459

Both the list of files and the definition of trials are available upon request.

For the alternative way of training which will be detailed in Section 4.1, we needed to prepare a *simulated training dataset*. The source LibriSpeech recordings were corrupted to obtain a dataset that equals the multichannel VOiCES training set in size. It means that utterances in the simulated and retransmitted training sets are the same. The difference is in a channel (real vs. simulated). To account for conditions in the VOiCES corpus, room simulation (employing image source method [20]) and noise addition were performed. Reverberation time RT60 of imaginary rooms was randomly drawn from interval [0.3, 0.9] s. To each of them, we randomly placed sources of speech and noise and also four omnidirectional microphones. Noises were selected from the Freesound library[4] and contain shop, crowd, library, office, real fan and street sounds that are added with SNRs ranging from 3 to 20 dB.

3.2. X-vector extractor architecture and re-training

As the application-specific model, we use the x-vector extractor corresponding to model 14 in Table 2 in [4] trained with the Kaldi toolkit [21]. This model is based on the `SRE16/v2` recipe with several modifications. Instead of the architecture in the recipe, we use the deeper architecture proposed in [22]. This architecture has nine TDNN/DNN layers in the frame-wise block, resulting in a total context of 11 frames on each side of a center frame. The block after pooling has two DNN layers. See Table 2 for details. The x-vector DNN was trained on 1.2 million speech segments from 7,146 speakers from the VoxCeleb 1 and 2 development data sets plus additional 5 million segments obtained with data augmentation. All training segments were 200 frames long and the model was trained for 9 epochs. Its performance on single channel VOiES development and evaluation sets is displayed in Table 1.

The x-vector extractor network is trained to classify speakers in the training set (Voxceleb in our case). Eventually, we require a classifier error to be propagated all the way to the mask estimator. However, the VOiCES multichannel dataset, used for the speech enhancer training, consists of a different set of speakers. In order to keep good performance of the extractor and enable it to classify 200 VOiCES speakers, we replaced the final linear layer and retrained it while

[4]http://www.freesound.org

Table 2. x-vector topology proposed in [22]. 30-dimensional MFCC features are inputs to the network, T is the number of training segment frames, and N is the number of speakers.

Layer	Layer context	(Input) × output
frame1	$[t-2, t-1, t, t+1, t+2]$	$(5 \times 30) \times 512$
frame2	$[t]$	512×512
frame3	$[t-2, t, t+2]$	$(3 \times 512) \times 512$
frame4	$[t]$	512×512
frame5	$[t-3, t, t+3]$	$(3 \times 512) \times 512$
frame6	$[t]$	512×512
frame7	$[t-4, t, t+4]$	$(3 \times 512) \times 512$
frame8	$[t]$	512×512
frame9	$[t]$	512×1500
stats pooling	$[0, T)$	1500×3000
segment1	0	3000×512
segment2	0	512×512
softmax	0	$512 \times$ N

keeping the rest of the parameters frozen. Since x-vectors are extracted from one of the preceding layers, this modification has no effect on them.

3.3. Mask estimator architecture

In this work, our aim is to introduce a proof-of-concept model, therefore we did not perform extensive architecture exploration. The network topology resembles the one employed in [13] with minor modifications. The input to the network is the magnitude spectrum. The input time-domain signal with a sampling frequency of 16 kHz is transformed to STFT domain using frames of 1024 samples and 256 samples frame-shift. The output is decoupled into two parts as shown in Figure 1 – one output serves to mask out speech and another one to mask out noise. Both have a dimension of 513 to match the input spectrum. The architecture comprises the following layers:

LSTM layer 256 units, tanh activation function, $p = 0.5$ dropout

linear layer 513 units, sigmoid activation function, $p = 0.5$ dropout

linear layer 513 units, sigmoid activation function, $p = 0.5$ dropout

linear output layer 2×513 units, sigmoid activation function

3.4. Speaker verification backend

We used an identical backend to the one in the Kaldi x-vector recipe. This backend involves a preprocessing step which first reduces the x-vector dimension by LDA from 512 to 250, and then applies a length-normalization. For training the backend, we concatenated all segments from each session of the VoxCeleb 1 and 2 development data. Including augmentations, this resulted in 830K files.

4. EVALUATION

In order to assess our proposed framework, we employ the BeamfomIt toolkit [23] as a baseline multichannel beamformer. It does not fit the framework in a sense that it has no learnable parameters but it is well established in the speech community. Speaker verification performance obtained with this fixed beamformer is displayed in the first row of Table 3. The results are expressed in terms of equal error rate (EER [%]) and minimum detection cost (C_{det}) as defined for the VOiCES challenge in [19] (prior target probability P_{tar} is set to 0.01).

4.1. Separate mask estimator training

In order to see the effect of application-aware training, we first train the mask estimator separately and plug it into the processing chain. This can also be viewed as another baseline that is more difficult to overcome. The results of independent training will then also be used as seed models and trained further with speaker verification related objective. We explored two approaches to mask estimator learning as follows.

Masks optimization

The first approach to training is inspired by [13]. The network is trained to estimate so-called ideal binary masks (IBM). The objective is a minimization of binary cross-entropy (BCE) between values in estimated masks and IBMs that are either 0 or 1. IBMs reflect the dominance of speech or noise in each time-frequency bin. Therefore, the knowledge of speech and additive noise components in the input recording is required for training. This is not available for the VOiCES corpus because the data are retransmitted which also introduces a convolutive distortion and one cannot easily obtain noisy component subtracting the source clean recording from the replayed recording. Therefore, the *simulated training dataset* was used in this experiment. In order to satisfy the requirement of having an exact knowledge of speech and additive noise components, as follows from (2), we convolve the original Librispech recordings with the first 50 ms of the generated room impulse responses (RIR) and obtain a clean component of the audio. By performing convolution of the source with the rest of RIR and addition of reverberant noise, the noise component is obtained. Both components are used to compute IBMs and their sum provides a noisy and reverberant signal as recorded by an imaginary microphone.

Beamformer output optimization

Since we have solved propagation through generalized eigenvalue decomposition, we are able to train the mask estimator to directly optimize the ability of the beamformer to recover the clean signal at its output. In this way of training,

we append the first group of components from Figure 1 to the mask estimator and use mean squared errror (MSE) between the magnitude STFT output of the beamformer and clean source speech (from LibriSpeech) as a loss function. This approach is more convenient since it does not depend on an exact knowledge of additive noise (which is usually available only for simulated data). Therefore, we used the multichannel VOiCES training dataset for training. We briefly experimented with using simulated data for MSE loss optimization, but currently we achieve significantly worse results compared to training with the multichannel VOiCES set.

We term the two approaches *separate* in Table 3. As observed, beamformers using mask estimators obtained by performing application-unaware training outperform baseline BeamformIt in either of the metrics on development as well as evaluation sets. BCE optimizing estimator produces better results in all metrics compared to the baseline with significant gain in terms of C_{det}. It yields better performance compared to the MSE optimizing approach. It might be because it fits the definition of the GEV beamformer which requires noise and speech PSD matrices computed using masks. In the latter approach, the network needs to learn to estimate proper noise and speech masks to perform beamforing which is presumably more difficult task.

4.2. Application-aware mask estimator training

Finally, we performed training of the beamformer's mask estimator in our framework optimizing the application-specific objective – cross-entropy (CE) of the x-vector extractor. The training may start from a random initialization of mask estimator's parameters. The speech enhancement component thus needs to learn a reasonable combination of channels in a beamformer framework solely based on the classification error. This experiment is referred to as *app-aware, scratch*. We also made use of models pre-trained with approaches described in 4.1 instead of starting from scratch. Those were further optimized using a CE objective and we refer to them as *app-aware, pretr* in Table 3.

When training the beamformer from a random initialization, it has to deal with a difficult task of figuring out that beamformer requires noise and speech PSD matrices (as well as the *separate, MSE* model). However, compared to training the mask estimator with the MSE objective, speaker verification loss helps it to reach better performance. Despite its superiority over MSE objective based model, it still does not reach the performance of beamformer independently trained with the BCE objective.

In the case when the *separate, MSE* model is used as a seed model for the CE training, an unstable behavior is observed. It may correspond with the fact that MSE trained front-end did not learn to do proper beamforming and rather learned to emphasize voices in the training set (that are also in the development set). This initialization is a good starting point to decrease metrics of our interest on the development set rapidly during training. However, the model was unable to generalize well. In the initial phase of training, error rates on the evaluation set had increased and then they started to decrease, but the performance of the seed model was not surpassed. This suggests that the MSE pre-training is not ideal.

On the other hand, BCE optimized mask estimator is a promising seed model. The separate training forced it to support real beamforming but IBMs, that model tries to predict, are handcrafted which suggests room for improvement. It was achieved via training in the proposed framework.

Table 3. Results of application-aware and application-unaware beamforming in terms of equal error rate (EER [%]) and minimum detection cost C_{det}. *BF obj.* stands for the objective function used for a beamformer mask estimation network training: CE – cross-entropy (multi-class), BCE – binary cross-entropy, MSE – mean squared error. Plus sign separates the objective used for pre-training and the application-aware training.

Method	**BF obj.**	**Dev. set**		**Eval. set**	
		EER	C_{det}	**EER**	C_{det}
BeamformIt	–	1.73	0.407	5.11	0.494
App-aware, scratch	CE	1.77	0.202	4.39	0.456
Separate	MSE	1.91	0.207	4.77	0.497
App-aware, pretr.	MSE + CE	1.13	0.146	4.78	0.496
Separate	BCE	1.53	0.214	4.26	0.446
App-aware, pretr.	BCE + CE	1.37	0.162	4.02	0.417

5. CONCLUSIONS

In this work, we have succeeded in improving the speaker verification system in the domain of data captured with distant microphone arrays via application-aware training of the underlying beamformer. We have shown that application-aware beamformer re-training outperforms application independent training with just the BCE objective. We have also shown that using a randomly initialized beamformer and performing only application-aware training still produces good results and outperforms the baseline system built on top of the BeamformIt toolkit. Even though not completely successfully, we have introduced training of the beamforming mask estimator via MSE between the original and beamformed signal which allows for simpler training and invites for more future work with the aim towards robustness and generalization on unseen speakers. We can also consider multi-task training by adding the MSE loss to the cross-entropy objective in the very end of the model as well as joint training of the beamformer and the x-vector extractor.

6. REFERENCES

[1] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep Neural Network Embeddings for Text-Independent Speaker Verification," in *Interspeech 2017*, Aug 2017.

[2] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-End Factor Analysis for Speaker Verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011, ISSN: 15587916.

[3] S. J. D. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *Proc. International Conference on Computer Vision (ICCV)*, Rio de Janeiro, Brazil, 2007.

[4] P. Matějka, O. Plchot, H. Zeinali, L. Mošner, A. Silnova, L. Burget, and O. Glembek, "Analysis of BUT submission in far-field scenarios of VOiCES 2019 challenge," in *Interspeech 2019*, Sep 2019.

[5] D. S. Kulkarni, R. R. Deshmukh, and P. P. Shrishrimal, "A Review of Speech Signal Enhancement Techniques," *International Journal of Computer Applications*, vol. 139, no. 14, pp. 23–26, 2016.

[6] I. Himawan, I. McCowan, and S. Sridharan, "Clustered Blind Beamforming From Ad-Hoc Microphone Arrays," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 661–676, May 2011.

[7] J. Capon, "High-resolution frequency-wavenumber spectrum analysis," *Proceedings of the IEEE*, vol. 57, no. 8, pp. 1408–1418, 1969.

[8] J. Heymann, L. Drude, and R. Haeb-Umbach, "Wide Residual BLSTM Network with Discriminative Speaker Adaptation for Robust Speech Recognition," in *Proceedings of the 4th International Workshop on Speech Processing in Everyday Environments (CHiME16)*.

[9] B. Li, T. N. Sainath, R. J. Weiss, K. W. Wilson, and M. Bacchiani, "Neural Network Adaptive Beamforming for Robust Multichannel Speech Recognition," in *Proceedings of Interspeech*, 2016.

[10] T. N. Sainath, R. J. Weiss, K. W. Wilson, A. Narayanan, and M. Bacchiani, "Factored spatial and spectral multichannel raw waveform CLDNNs," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016.

[11] T. N. Sainath, R. J. Weiss, K. W. Wilson, B. Li, A. Narayanan, E. Variani, M. Bacchiani, I. Shafran, A. Senior, K. Chin, A. Misra, and C. Kim, "Multichannel Signal Processing with Deep Neural Networks for Automatic Speech Recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, May 2017.

[12] X. Xiao, S. Watanabe, H. Erdogan, L. Lu, J. Hershey, M. L. Seltzer, G. Chen, Y. Zhang, M. Mandel, and D. Yu, "Deep beamforming networks for multi-channel speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016, pp. 5745–5749.

[13] J. Heymann, L. Drude, and R. Haeb-Umbach, "Neural network based spectral mask estimation for acoustic beamforming," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2016, pp. 196–200, IEEE.

[14] J. Heymann, L. Drude, C. Boeddeker, P. Hanebrink, and R. Haeb-Umbach, "Beamnet: End-to-end training of a beamformer-supported multi-channel ASR system," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2017.

[15] C. Richey, M. A. Barrios, Z. Armstrong, C. Bartels, H. Franco, M. Graciarena, A. Lawson, M. K. Nandwana, A. Stauffer, and J. van Hout, "Voices Obscured in Complex Environmental Settings (VOICES) corpus," *arXiv e-prints*, p. arXiv:1804.05053, Apr 2018.

[16] E. Warsitz and R. Haeb-Umbach, "Blind Acoustic Beamforming Based on Generalized Eigenvalue Decomposition," *IEEE Transactions on Audio, Speech, and Language Processing*, July 2007.

[17] H. Erdogan, J. R. Hershey, S. Watanabe, M. I. Mandel, and J. Le Roux, "Improved MVDR Beamforming Using Single-Channel Mask Prediction Networks," pp. 1981–1985.

[18] C. Boeddeker, P. Hanebrink, L. Drude, J. Heymann, and R. Haeb-Umbach, "Optimizing neural-network supported acoustic beamforming by algorithmic differentiation," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2017, pp. 171–175, IEEE.

[19] M. K. Nandwana, J. van Hout, M. McLaren, C. Richey, A. Lawson, and M. A. Barrios, "The VOiCES from a Distance Challenge 2019 Evaluation Plan," *arXiv e-prints*, p. arXiv:1902.10828, Feb 2019.

[20] J. B. Allen and D. A. Berkley, "Image method for efficiently simulating small-room acoustics," *Journal of the Acoustical Society of America*, vol. 65, no. 4, pp. 943–950, 1979, ISSN: 0001-4966.

[21] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, et al., "The Kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*. IEEE Signal Processing Society, 2011, number EPFL-CONF-192584.

[22] D. Snyder, D. Garcia-Romero, G. Sell, A. McCree, D. Povey, and S. Khudanpur, "Speaker Recognition for Multi-speaker Conversations Using x-vectors," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019, pp. 5796–5800.

[23] X. Anguera, C. Wooters, and J. Hernando, "Acoustic beamforming for speaker diarization of meetings," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 7, pp. 2011–2021, September 2007.