# RESOURCE EFFICIENT DEEP EIGENVECTOR BEAMFORMING

*Matthias Zöhrer*[1], *Lukas Pfeifenberger*[1], *Günther Schindler*[2], *Holger Fröning*[2], *Franz Pernkopf*[1]

[1]Signal Processing and Speech Communication Lab
Graz University of Technology
[2]Institute of Computer Engineering
University of Heidelberg

## ABSTRACT

We propose binary neural networks (BNNs) for acoustic beamforming. This makes the speech enhancement approach resource efficient and applicable for embedded applications. Using CHiME4 data, we use BNNs to estimate the speech presence probability mask for GEV-PAN beamformers. By doing so, we achieve audio quality and ASR scores on par to single-precision deep neural networks (DNNs), while the computational requirements and the memory footprint are significantly reduced.

***Index Terms***— GEV-PAN beamformer, Binary Neural Network, Reduced Precision, Speech Enhancement

## 1. INTRODUCTION

Deep neural networks (DNNs) achieve impressive performances in many applications such as computer vision [1], speech recognition [2], and machine translation [3], among others. This is particularly true when having big amounts of data and almost unlimited computing resources available. However, in real-world scenarios the computing infrastructure is often limited.

When having only computing architectures with limited resources available, DNNs cannot run efficiently anymore, e.g. low-cost embedded hardware often does not have sufficient amount of memory, power or processing units to run DNNs efficiently. There is an emerging trend in developing NN architectures that can be evaluated in a fast and energy-efficient way requiring only little memory for the parameters. Several efforts have been made to reduce the number of bits required to store the weights which usually results in faster computation and a smaller memory footprint. Recently, methods have been proposed which learn double precision weights and then reduce the number of bits until the classification performance starts to degrade [4]. This direction of research has been pushed towards NNs that require in the extreme case only a single bit per weight, i.e. expensive floating point operations reduce to simple bit operations [5, 6]. One approach to train NNs with discrete weights and activation functions, whose derivative is zero almost everywhere, is based on the straight through gradient estimator [7]. Using this straight through estimator allows to train DNNs with binary weights [5, 8]. The use of binary weights dramatically reduce the computational requirements and memory footprint, and hence support to use the DNNs on resource constrained architectures.

This work was supported by the Austrian Science Fund (FWF) under the project number I2706-N31 and NVIDIA for providing GPUs.

Recently, in several speech enhancement applications single-precision DNNs have been used to outperform traditional model-based algorithms. Appreciable examples in the field of acoustic beamforming include [9–12], where a DNN estimates the speech mask which is used to determine the power spectral density (PSD) matrices of the multi-channel speech and noise signals. With these PSD matrices a beamforming filter such as the minimum variance distortionless response (MVDR) beamformer or generalized Eigenvector (GEV) beamformer can be obtained. In [12], we proposed deep eigenvector beamforming, where we used the dominant eigenvector of the noisy speech PSD matrix as feature vector and DNNs to estimate the speech mask. Those DNNs achieve impressive performances, but these models are inefficient when it comes to computational and memory requirements. In a recent paper we reduce the computational requirements of our speech mask estimator using logistic regression [13]. This approach reduces the number of parameters by a factor of 100.

In this paper, we go a step further. We propose DNNs with binary weights for the estimation of the speech mask. In particular, the time consuming multiplication of input $x$ by weight matrix $W$ in a DNN is replaced by XNOR operations. This reduces the computational requirements and memory footprint and we are able to run the DNN on small embedded architectures. In our experiments we observe that the estimation of the speech mask by binary neural networks (BNNs) for the GEV beamformer achieves consistent audio quality and ASR scores compared to a single-precision DNN baseline, using CHiME4 data. Furthermore, our BNN enables a computation speedup by a factor of 5.7 and 11.7 on a GPU and ARM processor, respectively.

This work is organized as follows: Section 2 introduces deep GEV beamformers. In Section 3, BNNs for speech mask estimation are discussed. Section 4 lists experimental results. In particular, the speech mask accuracy, the perceptual audio quality and speech recognition results of the GEV beamformer are shown. Furthermore, the computational complexity of BNNs is evaluated on ARM, GPU and field programmable gate arrays (FPGAs) and compared to single-precision DNN baselines. Section 5 concludes the paper.

## 2. DEEP EIGENVECTOR BEAMFORMING

In our beamforming setup, we assume a single speech source embedded in ambient noise. The array consists of $M$ microphones, arranged into an arbitrary array geometry. Figure 1 shows the main components of the deep eigenvector beamformer.

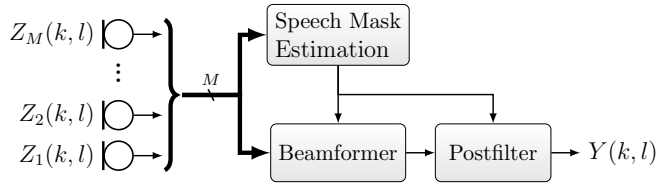


**Fig. 1**. System overview, showing the microphone signals $Z_m(k,l)$ and the beamformer+postfilter output $Y(k,l)$ in frequency domain.

The deep eigenvector beamformer employs the GEV beamformer [14, 15] and the phase-aware-normalization (PAN) postfilter introduced in [13].

The beamformer requires a speech mask $\mathrm{p}_{SPP}(k,l)$ for each frequency bin $k$ and time frame $l$. This mask is estimated from the dominant eigenvector of the noisy speech PSD $\mathbf{\Phi}_{ZZ}$, which is obtained using recursive averaging: $\mathbf{\Phi}_{ZZ}(k,l) = \mathbf{\Phi}_{ZZ}(k,l-1)\alpha + (1-\alpha)\boldsymbol{Z}(k,l)\boldsymbol{Z}^H(k,l)$, where $0 \leq \alpha \leq 1$ is a smoothing parameter and $\mathbf{Z}(k,l) = [Z_1(k,l), ..., Z_M(k,l)]^H$.
The eigenvalue decomposition of the noisy speech PSD matrix gives $\mathbf{\Phi}_{ZZ}(k,l) = \sum_{m=1}^{M} \lambda_{Z_m}(k,l)\boldsymbol{v}_{Z_m}(k,l)\boldsymbol{v}_{Z_m}^H(k,l)$, where $\lambda_{Z_m}(k,l)$ and $\boldsymbol{v}_{Z_m}(k,l)$ are its eigenvalues and eigenvectors, respectively [16].

The DNN used for speech mask estimation uses the cosine similarity between neighboring eigenvectors, i.e.

$$x_{\Delta}(k,l) = |\boldsymbol{v}_{Z_1}(k,l)^H \boldsymbol{v}_{Z_1}(k,l-\Delta)| \tag{1}$$

as input features, where $\boldsymbol{v}_{Z_1}(k,l)$ denotes the dominant eigenvector of $\mathbf{\Phi}_{ZZ}(k,l)$. Note that each feature $x_{\Delta}(k,l)$ compresses the information of all $M$ channels into a scalar value between 0 and 1, it is therefore independent of the signal energy and the number of microphones being used in the setup. To observe a significant difference between two neighboring eigenvectors, the matrix $\mathbf{\Phi}_{ZZ}(k,l)$ has to be updated with a sufficiently small time constant. During speaker activity, $x_{\Delta}(k,l)$ is close to one, and close to zero in case of undirected ambient noise. Further details of GEV-PAN can be found in [13].

## 3. TRAINING DNNS USING BINARY WEIGHTS

We replace the DNN used for mask estimation with a BNN [5, 17–19]. This makes the speech enhancement approach resource efficient.

A single layer of a BNN is shown in Figure 2.


**Fig. 2**. BNN layer.

Instead of multiplying any arbitrary input $\underline{\mathrm{x}}(k,l)$ by the weights $\underline{\mathrm{W}}$, a BNN uses a single XNOR operation for each input and accumulates the output. Then batch-normalization [20] and a non-linear activation function is applied. Due to binary weights the *sign* function is used as activation. Batch-normalization is efficiently computed by combining thresholding with the *sign* activation function [21], removing the need for a float32 operation within the network layer. This specific structure of the network makes the model very effective in terms of computational needs and memory. The network operates in integer range and can be effectively ported to a DSP or FPGA architecture.

When embedding BNNs in a GEV beamformer, we have to scale the inputs and outputs of the NN to an appropriate range. Therefore, we propose a simple and effective conversion of the eigenvector features to the integer range. Given single-precision eigenvector features $x_{\Delta}(k,l)$ in the range between zero and one, we scale $x_{\Delta}(k,l)$ to 8bit integer values in the range +-127. After applying the BNN, we rescale the network's output to the output range of the $\mathrm{p}_{SPP}$, i.e $\{0,1\} \in \mathbb{R}$ and obtain the estimated speech presence probability mask. This deep binary speech mask estimator is shown in Figure 3.

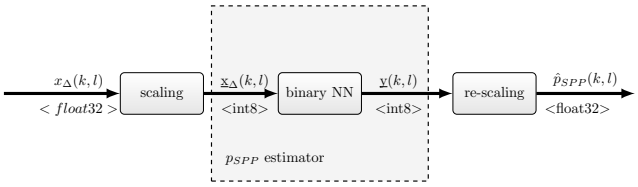


**Fig. 3**. Binary $\mathrm{p}_{SPP}$ estimator.

Training a BNN is more difficult [5]. In particular, we replace *xnor()* and *sign()* functions by common single-precision multiplications and *tanh* activations, rounded to the range $\{-1,1\}$. For the forward-propagation process, we round the continuous weights $\{W_1, ..., W_L\} \in \mathbb{R}$ to binary weights $\theta_b = \{\underline{\mathrm{W}}_1, ..., \underline{\mathrm{W}}_L\} \in \{-1,1\}$ and compute a forward-path over multiple layers using a linear output layer. Clipping the accumulated values of the vector-matrix products to a fixed bit width of eight further reduces the complexity of the NN and prevents overflows. After obtaining an output activation with the binary forward-path, we determine the MSE and propagate the gradient through the network (straight through estimation) [5]. Non-differentiable functions are linearly approximated and the gradient is determined for continuous weights, which are updated. Further details can be found in [5, 21].

## 4. EXPERIMENTAL RESULTS

### 4.1. Database

To evaluate the binary deep GEV beamformer, we use the CHiME4 corpus [22]. CHiME4 provides 2 and 6-channel recordings of a close-talking speaker corrupted by four different types of ambient noise. The database consists of a training set (tr05), a validation set (dt05) and a test set (et05). Ground truth utterances (i.e. the separated speech and noise signals) are available for all recordings and the true speech masks $\hat{\mathrm{p}}_{SPP,opt}(k,l)$ can be computed [16]. Once trained, the deep binary GEV beamformer provides a prediction $\mathrm{p}_{SPP}(k,l)$ for each utterance; required to calculate $\mathbf{\Phi}_{SS}(k,l)$ and $\mathbf{\Phi}_{NN}(k,l)$ [23]. We use a STFT window length of 32ms for determining $\mathbf{\Phi}_{ZZ}(k,l)$ and an overlap of 50% to process the data. The averaging window length for PSD estimation is T = 250ms. The speech and noise PSD estimates are used to construct the GEV-PAN beamformer. We use all utterances (real and simu) from the train (tr05), the validation (dt05) and test set (et05).

### 4.2. Evaluating the Speech Mask Accuracy

We used a fully connected 3-layer BNN with a linear output layer for all 2 and 6 channel experiments. The cosine similarity $x_\Delta$ was determined for $\Delta$=1. The BNN was trained using ADAM [24] with default parameters using framewise inputs $x_{\Delta=1}(l) = [x_{\Delta=1}(1,l), ..., x_{\Delta=1}(k,l)]^T$ and the speech mask probabilities $\mathrm{p}_{SPP}$. Dropout with a probability of 0.25 was used during training. We used a single-precision 3-layer network with 513 neurons per layer and BNNs with 513 and 1024 neurons per layer.

| model | neurons / layer | channels | **train** | **valid** | **test** |
|---|---|---|---|---|---|
| DNN | 513 | 2ch | 5.8 | 6.2 | 7.7 |
| BNN | 513 | 2ch | 6.2 | 6.2 | 7.9 |
| BNN | 1024 | 2ch | 6.2 | 6.6 | 7.9 |
| DNN | 513 | 6ch | 4.5 | 3.9 | 4.0 |
| BNN | 513 | 6ch | 4.7 | 4.1 | 4.4 |
| BNN | 1024 | 6ch | 4.9 | 4.2 | 4.1 |

**Table 1**. Mask prediction error $\mathcal{L}$ in %.

Table 1 reports the mask predictions error

$$\mathcal{L} = \frac{100}{KL} \sum_{k=1}^{K} \sum_{l=1}^{L} \left| \hat{\mathrm{p}}_{SPP}(k,l) - \mathrm{p}_{SPP,opt}(k,l) \right|. \qquad (2)$$

Single precision networks achieved the best prediction error. The BNNs achieve comparable results. Doubling the networks size of BNNs slightly improved the error on the test set.

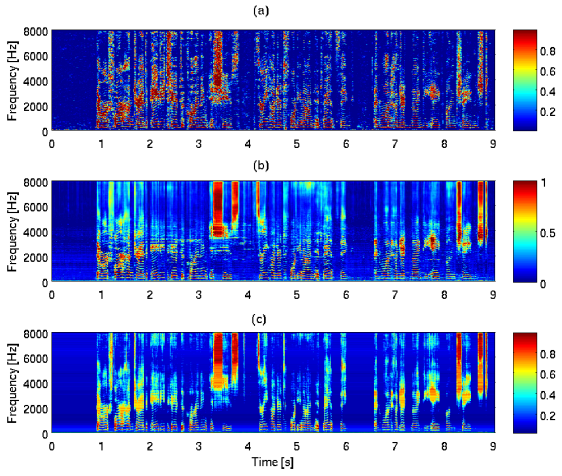


**Fig. 4**. (a) True speech mask $\mathrm{p}_{SPP,opt}(k,l)$; (b) prediction of $\hat{\mathrm{p}}_{SPP}(k,l)$ by DNN (c) prediction of $\hat{\mathrm{p}}_{SPP}(k,l)$ by BNN.

Figure 4 shows the speech mask of the BNN for the utterance F01_22HC010W_BUS. Panel (a) shows the optimal speech mask $\mathrm{p}_{SPP,opt}$, and panel (b) and (c) show the predicted speech masks for the DNN with 513 neurons/layer and the BNN with 1024 neurons/layer, respectively.

### 4.3. Evaluating the Perceptual Audio Quality

Given the predicted speech mask $\hat{\mathrm{p}}_{SPP}(k,l)$, we construct the GEV-PAN beamformer [25] for both the 2 and 6-channel data. The *Overall Perceptual Score* (OPS) [26] and PESQ [27] were used to evaluate the performance of the resulting speech signal $Y(k,l)$ in terms of perceptual speech quality. Ground truth estimates required for these scores are obtained using the $\mathrm{p}_{SPP,opt}(k,l)$ and the GEV-PAN.

| method | set | **train** | **valid** | **test** |
|---|---|---|---|---|
| CHiME4 baseline | simu | 1.35 | 1.31 | 1.26 |
| (BeamformIt), 5ch [22] | real | 1.35 | 1.28 | 1.37 |
| CGMM-EM with MVDR | simu | 1.79 | 1.59 | 1.51 |
| and postfilter, 6ch [28] | real | 1.53 | 1.41 | 1.44 |
| **DNN (513 neurons / layer)** | **simu** | **2.16** | **2.12** | **2.35** |
| **with GEV-PAN, 2ch** | **real** | **2.14** | **2.02** | **1.94** |
| **BNN (513 neurons / layer)** | **simu** | **2.00** | **1.97** | **2.10** |
| **with GEV-PAN, 2ch** | **real** | **1.82** | **1.82** | **1.60** |
| **BNN (1024 neurons / layer)** | **simu** | **2.00** | **2.19** | **2.59** |
| **with GEV-PAN, 2ch** | **real** | **2.16** | **2.10** | **2.07** |
| **DNN (513 neurons / layer)** | **simu** | **2.54** | **2.57** | **2.94** |
| **with GEV-PAN 6ch** | **real** | **2.47** | **2.41** | **2.33** |
| **BNN (513 neurons / layer)** | **simu** | **2.08** | **1.97** | **2.17** |
| **with GEV-PAN, 6ch** | **real** | **2.00** | **2.51** | **1.99** |
| **BNN (1024 neurons / layer)** | **simu** | **2.04** | **2.05** | **2.24** |
| **with GEV-PAN, 6ch** | **real** | **2.00** | **1.93** | **1.71** |

**Table 2**. PESQ scores.

Table 2 lists the PESQ scores of both DNNs and BNNs using the GEV-PAN beamformer. All models achieved solid PESQ scores on all datasets, outperforming the CHiME4-baseline enhancement system, i.e. the BeamformIt!-toolkit [22], and the front-end of the best CHiME3 system [28], i.e. CGMM-EM. The 2-channel BNN with 1024 neurons/layer obtained a slightly better score than its single-precision DNN baseline. Using 6-channel data, the single-precision DNN achieved the best result.

| method | set | **train** | **valid** | **test** |
|---|---|---|---|---|
| CHiME4 baseline | simu | 33.11 | 34.73 | 31.46 |
| (BeamformIt), 5ch [22] | real | 29.97 | 36.45 | 36.74 |
| CGMM-EM with MVDR | simu | 52.15 | 43.02 | 40.59 |
| and postfilter, 6ch [28] | real | 44.95 | 41.89 | 36.87 |
| **DNN (513 neurons / layer)** | simu | **64.21** | **61.74** | **56.32** |
| **with GEV-PAN, 2ch** | real | **64.21** | **62.72** | **56.32** |
| BNN (513 neurons / layer) | simu | 58.11 | 57.58 | 57.58 |
| with GEV-PAN, 2ch | real | 56.79 | 57.52 | 41.24 |
| **BNN (1024 neurons / layer)** | simu | **61.64** | **60.78** | **54.20** |
| **with GEV-PAN, 2ch** | real | **61.64** | **60.78** | **45.22** |
| **DNN (513 neurons / layer) ,** | simu | **67.98** | **66.76** | **68.71** |
| **with GEV-PAN 6ch** | real | **69.98** | **70.33** | **63.28** |
| BNN (513 neurons / layer) with | simu | 61.44 | 55.87 | 62.39 |
| with GEV-PAN, 6ch | real | 63.03 | 64.77 | 64.52 |
| **BNN (1024 neurons / layer)** | simu | **65.59** | **64.98** | **68.41** |
| **with GEV-PAN, 6ch** | real | **67.91** | **68.41** | **59.94** |

**Table 3**. OPS scores.

Table 3 reports the OPS given the enhanced utterances using 2-channel and 6-channel data. Again good results are achieved using the GEV-PAN beamformer. Doubling the network size of BNNs mostly improves the OPS scores. In general, BNNs achieve on average a slightly lower OPS score than the single-precision DNN baseline, on both 2-channel and 6-channel data.

### 4.4. Evaluation by Automatic Speech Recognition

Table 4 reports the word error rate (WER) for the 2 and 6 channel data using DNNs and BNNs with a GEV-PAN beamformer. Google's speech recognition service[1] was used to obtain WERs in all experiments. Groundtruth transcriptions were generated using original WSJ0 recordings.

| method | set | **train** | **valid** | **test** |
|---|---|---|---|---|
| DNN (513 neurons / layer) | simu | 10.47 | 15.25 | 25.61 |
| with GEV-PAN, 2ch | real | 27.56 | 12.04 | 27.56 |
| BNN (513 neurons / layer) | simu | 12.13 | 16.52 | 31.24 |
| with GEV-PAN, 2ch | real | 26.17 | 16.26 | 35.38 |
| BNN (1024 neurons / layer) | simu | 11.01 | 15.58 | 27.56 |
| with GEV-PAN, 2ch | real | 24.69 | 15.56 | 30.62 |
| **DNN (513 neurons / layer)** | **simu** | **9.80** | **13.66** | **15.31** |
| **with GEV-PAN, 6ch** | **real** | **23.79** | **9.58** | **18.68** |
| BNN (513 neurons / layer) | simu | 11.59 | 15.29 | 16.56 |
| with GEV-PAN, 6ch | real | 24.7 | 13.57 | 21.18 |
| BNN (1024 neurons / layer) | simu | 10.71 | 14.78 | 16.11 |
| with GEV-PAN, 6ch | real | 24.01 | 13.40 | 20.29 |

**Table 4**. WER scores for our experiments.

Interestingly, for both DNNs and BNNs the WER of the 2 channel test utterances is large. However, when using 6 microphones, the WERs on both simulation and real test set are significantly improved. In general, BNNs achieve similar results compared to DNNs. In case of 6-channel experiments, the best WER was obtained by DNNs.

### 4.5. Evaluating the Computational Complexity

**General-Purpose Processors:** In order to show the advantages of binary computation on general-purpose processors, we implemented matrix-multiplication operators for NVIDIA GPUs and CPUs. From computational perspective, the classification of BNNs can be implemented very efficiently as binary-scalar products. Matrix multiplications can be computed by bit-wise *xnor()* operation, followed by counting the number of set bits with *popc()*:

$$x * y = N - 2 * popc(xnor(x, y)), x_i, y_i \in [-1, +1] \forall i \quad (3)$$

We use the matrix-multiplication algorithms of the MAGMA and Eigen library and replace float multiplications by *xnor()* operation, as depicted in Equation 3. Our CPU implementation uses NEON vectorization in order to fully exploit SIMD units on ARM processors. We report execution time of GPU and ARM CPU in Table 5. We don't report performance results of x86 architectures because neither SSE nor AVX ISA supports vectorized *popc()*.

| arch | neurons | time (float32) | time (binary) | **speedup** |
|---|---|---|---|---|
| GPU | 256 | 0.14ms | 0.05ms | **2.8** |
| GPU | 513 | 0.34ms | 0.06ms | **5.7** |
| GPU | 1024 | 1.71ms | 0.16ms | **10.7** |
| GPU | 2048 | 12.87ms | 1.01ms | **12.7** |
| ARM | 256 | 0.42ms | 0.42ms | **8.7** |
| ARM | 513 | 1.43ms | 1.43ms | **11.7** |
| ARM | 1024 | 8.13ms | 8.13ms | **13.4** |
| ARM | 2048 | 771.33ms | 58.81ms | **13.1** |

**Table 5**. Performance metrics for matrix · matrix multiplications on a NVIDIA Tesla K80 and ARM Cortex-A57.

[1] https://developers.google.com/api-client-library/python/

As can be seen, binary calculations clearly outperform float32 calculation in terms of execution time. This also affects energy consumption since binary values require less off-chip accesses and operations.

**Specialized Processors:** Probably the most promising candidate for binary computations are FPGAs. Applying binarized neural networks on FPGAs has several advantages (compared to float, fixed-point, and integer networks) which enable faster classification, better energy efficiency, and less expensive hardware. Using logic operations for multiplications and *popc()* for accumulations results in higher resource efficiency because requirements on look up tables and flip flops decrease significantly. Further, Umuroglu et al. [21] showed that batch normalization and activation can be implemented efficiently by pre-computing thresholds and only using unsigned comparisons which reduces parameter size and resource requirements. Finally, using binary weights reduces parameter size by a factor of 32 - compared to float32 - which allows the use of cheaper hardware (model parameters have to be saved in the scarce block RAM). Table 6 compares parameter size and device costs (Xilinx Spartan-7 series) of the binary and float32 implementations.

| 3 layer DNN/BNN (513 neurons / layer) | | | | |
|---|---|---|---|---|
| Type | Par. Size | Device | Mem. Util. | **Cost (USD)** |
| float32 | 3.15 MB | XC7S100 | 73% | 181.10 |
| binary | 0.10 MB | XC7S6 | 56% | 22.33 |
| 3 layer DNN/BNN (1024 neurons / layer) | | | | |
| Type | Par. Size | Device | Mem. Util. | Cost (USD) |
| float32 | 7.35 MB | XC7S100 | 170% | 181.10 |
| binary | 0.23 MB | XC7S15 | 64% | 24.87 |

**Table 6**. Parameter and cost (retail cost Avnet) comparison for float32 and binary weights.

As shown in Table 6, the costs for chip requirements differ significantly between 32-bit and binary models. Further, the binary models show rather low memory utilization (and therefor more application potential) on the selected chips, whereas the 32-bit implementation of the larger DNN exceeds the largest chip of the Spartan-7 series.

## 5. CONCLUSIONS AND FUTURE WORK

We proposed neural networks with binary weights for acoustic beamforming. This makes the speech enhancement approach resource efficient and applicable for embedded systems. In particular, we used BNNs to estimate the speech mask $\mathrm{p}_{SPP}$. This mask allows to estimate the noise and speech power spectral density matrix used to determine the GEV-PAN beamformer. By doing so we achieve audio quality and ASR scores on par to single-precision DNNs, while the computational requirements and the memory footprint are significantly reduced. In future work, we aim to analyze tenary weights [29] for performance improvements.

## 6. REFERENCES

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., pp. 1097–1105. 2012.

[2] Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig, "Achieving human parity in conversational speech recognition," *CoRR*, vol. abs/1610.05256, 2016.

[3] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean, "Google's neural machine translation system: Bridging the gap between human and machine translation," *CoRR*, vol. abs/1609.08144, 2016.

[4] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David, "Training deep neural networks with low precision multiplications," *CoRR*, vol. abs/1412.7024, 2014.

[5] Matthieu Courbariaux and Yoshua Bengio, "Binarynet: Training deep neural networks with weights and activations constrained to +1 or -1," *CoRR*, vol. abs/1602.02830, 2016.

[6] Wolfgang Roth and Franz Pernkopf, "Discrete-valued neural networks using variational inference," Tech. Rep., Graz University of Technology, 2018.

[7] Yoshua Bengio, Nicholas Léonard, and Aaron C. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," *CoRR*, vol. abs/1308.3432, 2013.

[8] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David, "Binaryconnect: Training deep neural networks with binary weights during propagations," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds., pp. 3123–3131. 2015.

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

[10] J. Heymann, L. Drude, A. Chinaev, and R. Haeb-Umbach, "BLSTM supported GEV beamformer front-end for the 3rd chime challenge," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2015, pp. 444–451.

[11] H. Erdogan, J. Hershey, S Watanabe, M. Mandel, and J. L. Roux, "Improved MVDR beamforming using single-channel mask prediction networks," in *Interspeech*, 2016.

[12] L. Pfeifenberger, M. Zöhrer, and F. Pernkopf, "DNN-based speech mask estimation for eigenvector beamforming," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 66–70.

[13] L. Pfeifenberger, M. Zöhrer, and F. Pernkopf, "Eigenvector-based speech mask estimation using logistic regression," in *Interspeech*, 2017.

[14] E. Warsitz and R. Haeb-Umbach, "Blind acoustic beamforming based on generalized eigenvalue decomposition," in *IEEE Transactions on Audio, Speech, and Language Processing (ICASSP)*, 2007, vol. 15, pp. 1529–1539.

[15] E. Warsitz, A. Krueger, and R. Haeb-Umbach, "Speech enhancement with a new generalized eigenvector blocking matrix for application in a generalized sidelobe canceller," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2008, pp. 73–76.

[16] L. Pfeifenberger, M. Zöhrer, and F. Pernkopf, "Eigenvector-based speech mask estimation for multi-channel speech enhancement," in *Interspeech*, 2017.

[17] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," in *ECCV*, 2016.

[18] Cory P. Berg Alexander G. Anderson, "The high-dimensional geometry of binary neural networks," in *International Conference for Learning Representations (ICLR)*, 2017.

[19] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio, "Binarized neural networks," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds., pp. 4107–4115. Curran Associates, Inc., 2016.

[20] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *JMLR*, 2015, pp. 448–456.

[21] Yaman Umuroglu, Nicholas J. Fraser, Giulio Gambardella, Michaela Blott, Philip Leong, Magnus Jahre, and Kees Vissers, "Finn: A framework for fast, scalable binarized neural network inference," in *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, 2017, FPGA '17, pp. 65–74.

[22] Jon Barker, Ricard Marxer, Emmanuel Vincent, and Shinji Watanabe, "The third 'chime' speech separation and recognition challenge: Dataset, task and baselines," in *IEEE 2015 Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2015.

[23] M. Brandstein and D. Ward, *Microphone Arrays*, Heidelberg–New York: Springer Berlin,, 2001.

[24] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference for Learning Representations (ICLR)*, 2015.

[25] Lukas Pfeifenberger, Matthias Zöhrer, and Franz Pernkopf, "Dnn-based speech mask estimation for eigenvector beamforming," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2017, pp. 66–70.

[26] E. Vincent, "Improved perceptual metrics for the evaluation of audio source separation," *10th Int. Conf. on Latent Variable Analysis and Signal Separation (LVA/ICA)*, pp. 430–437, 2012.

[27] "ITU-T recommendation P.862. Perceptual evaluation of speech quality (PESQ): an objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs," 2000.

[28] T. Higuchi, N. Ito, T. Yoshioka, and T. Nakatani, "Robust MVDR beamforming using time-frequency masks for online/offline asr in noise," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2016, vol. 4, pp. 5210–5214.

[29] Fengfu Li, Bo Zhang, and Bin Liu, "Ternary weight networks," *arXiv preprint arXiv:1605.04711*, 2016.