INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

Keyword Spotting for Hearing Assistive Devices Robust to External Speakers

Iván López-Espejo[1], *Zheng-Hua Tan*[1], *Jesper Jensen*[1,2]

[1]Department of Electronic Systems, Aalborg University, Denmark
[2]Oticon A/S, Denmark

{ivl,zt,jje}@es.aau.dk, jesj@oticon.com

Abstract

Keyword spotting (KWS) is experiencing an upswing due to the pervasiveness of small electronic devices that allow interaction with them via speech. Often, KWS systems are speaker-independent, which means that any person —user or not— might trigger them. For applications like KWS for hearing assistive devices this is unacceptable, as only the user must be allowed to handle them. In this paper we propose KWS for hearing assistive devices that is robust to external speakers. A state-of-the-art deep residual network for small-footprint KWS is regarded as a basis to build upon. By following a multi-task learning scheme, this system is extended to jointly perform KWS and users' own-voice/external speaker detection with a negligible increase in the number of parameters. For experiments, we generate from the Google Speech Commands Dataset a speech corpus emulating hearing aids as a capturing device. Our results show that this multi-task deep residual network is able to achieve a KWS accuracy relative improvement of around 32% with respect to a system that does not deal with external speakers.

Index Terms: Robust keyword spotting, hearing assistive device, external speaker, multi-task learning

1. Introduction

Keyword spotting (KWS) aims at detecting a series of words from an audio stream comprising speech. This technology has become a popular research topic as it is considered a keystone for voice-based activation of virtual assistants (e.g., smart speakers) by means of keywords or wake-up-words [1].

Similarly, KWS may allow a hearing impaired person to initiate certain actions on her/his hearing assistive device, e.g., raising or lowering the volume. Since hearing aids are low computational resource devices, it is crucial that the KWS systems deployed on them have a small footprint (i.e., low memory and computational complexity).

Over the recent period, small-footprint KWS has attracted the attention of speech researchers due to the rapid development of deep learning [2]. For instance, in [3], Chen *et al.* proposed extracting keyword embeddings from a word-based long short-term memory (LSTM) acoustic model. During testing, successive embeddings are extracted from a sliding window and compared against keyword templates. This approach is shown to outperform a KWS system based on phoneme posteriorgram with dynamic time warping (DTW) [4].

Working towards end-to-end models is the trend in the context of small-footprint KWS. In [5], a deep neural network (DNN) is trained to predict keywords followed by a posterior handling technique outputting a confidence score. Robustness to both background noise and far-field conditions of this DNN-based KWS system is further improved in [6] through a combination of multi-style training and automatic gain control. Drawing from these works, the use of convolutional neural networks (CNNs) is explored in [7], which allows for improving DNN performance with far fewer parameters. This is a remarkable finding since maximizing KWS accuracy while having a compact model is essential in the context of low-resource devices. In this respect, it is not surprising that the number of both multiplications and parameters is directly correlated with the energy usage of the device that the KWS model is running on [8]. Finally, very recent work [9] investigates deep residual learning and dilated convolutions for small-footprint KWS and outperforms the previous state-of-the-art CNN-based system of [7] on the Google Speech Commands Dataset [10].

To the best of our knowledge, all of the above works are developed to be speaker-independent. This means that anyone should be able to trigger the proposed KWS systems. Nevertheless, for a number of applications we may want that only a particular speaker can interact with the system. This is often the case for hearing assistive devices, e.g., hearing aid systems, where the user must be the only one allowed to trigger actions on her/his hearing assistive device. Therefore, in this paper we explore KWS for hearing assistive devices that is robust to external speakers. Drawing from the deep residual network of [9], we consider multi-task learning to jointly perform KWS and own-voice/external speaker detection with a negligible increase in the number of parameters. For experimental purposes, a speech database emulating hearing aids as a capturing device is created from the Google Speech Commands Dataset. Our experimental results show that, thanks to exploiting the hearing aid multi-microphone signals, this multi-task deep residual network is able to improve KWS accuracy by about 32% compared to a system that does not deal with external speakers.

2. Deep Residual Learning for KWS

In this section we briefly review the deep residual network for small-footprint KWS proposed in [9] (`res15`), as this is regarded as a basis to build upon. This architecture is based on the work of He *et al.* [11], where the authors proposed residual learning to tackle the performance degradation that occurs when CNNs are too deep. Let $\mathbf{x}_{l-1}$ be the input to a particular layer l. For too deep networks, it might be easier to optimize the residual mapping $\mathcal{H}_l^{l+k}(\mathbf{x}_{l-1}) = \mathcal{F}_l^{l+k}(\mathbf{x}_{l-1}) + \mathbf{x}_{l-1}$ between layers l and $l+k$ ($k \in \mathbb{N}$) than the original mapping $\mathcal{F}_l^{l+k}(\mathbf{x}_{l-1})$ [11]. The above residual mapping can be simply accomplished by means of identity shortcut connections (identity mapping), i.e., those that skip $k+1$ layers.

The `res15` architecture [9], which uses Mel-frequency cepstral coefficients (MFCCs) as input, may be described as follows (see left part of Figure 1). The first layer is a convolutional layer, after which there is a total of six residual blocks with identity mapping. Every residual block comprises two additional convolutional layers each of them followed by a rectified

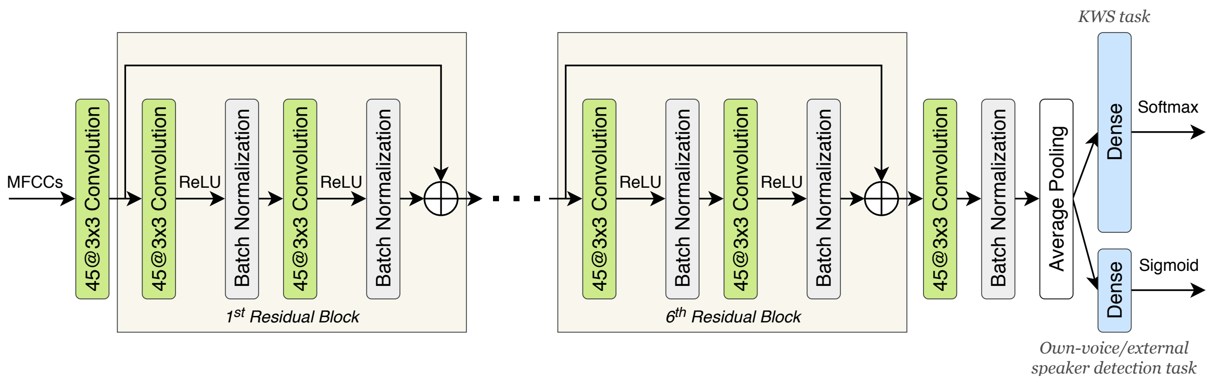

Figure 1: *Diagram of the multi-task deep residual network for KWS with own-voice/external speaker detection.*

linear unit (ReLU) activation function and a batch normalization layer. Convolutional layers in the residual blocks apply dilated convolutions with a dilation rate of $\left(2^{\lfloor \frac{l}{3} \rfloor}, 2^{\lfloor \frac{l}{3} \rfloor}\right)$, where $l = 0, ..., 11$ refers to the successive layers of this type and $\lfloor \cdot \rfloor$ denotes the floor function. Then, a non-residual convolutional layer with $(16, 16)$ convolution dilation, another batch normalization layer and an average pooling layer are appended to the deepest residual block. Finally, a fully-connected (dense) layer with softmax activation is used for keyword classification.

It should be noticed that for all the convolutional layers the bias vector is zero, the kernel size is 3×3 and the number of feature maps is set to 45. The reader is referred to [9] for further details on this deep residual network.

3. Multi-task Learning for KWS and Own-Voice/External Speaker Detection

We employ the state-of-the-art `res15` described in the previous section to perform KWS on hearing assistive devices. Let $\mathbf{X}$ be the input speech features to the model. In order to also let the deep residual network detect whether the user, S_u, or an external speaker, S_e, is trying to trigger the KWS system, we extend it with an additional output providing an estimate of the conditional probability $P(S_u|\mathbf{X}) = 1 - P(S_e|\mathbf{X})$. Then, keyword prediction from $\mathbf{X}$ is considered only if the estimate of $P(S_u|\mathbf{X})$ is above a certain threshold, e.g., 0.5.

When multi-task learning is considered for rather heterogeneous (i.e., dissimilar) tasks (as is our case), it makes sense that the task-specific output layers depend on different neuron activations. It is worth noticing that we conducted preliminary experiments by alternatively appending an average pooling layer and the own-voice/external speaker detection layer to the output of each residual block. However, these experiments revealed that there are no statistically significant differences in terms of both KWS and own-voice/external speaker detection accuracies as a function of the placement of the own-voice/external speaker detection layer. Therefore, in order to reduce the number of multiplications, we end up with the very simple solution to append such a layer, which consists of a fully-connected layer with one neuron and sigmoid activation, to the already existing average pooling layer, as can be seen in Figure 1.

We will show in Section 5 that this multi-task approach is highly effective to jointly perform KWS and own-voice/external speaker detection. We will also show that combining the two network outputs leads to significantly improved KWS scores.

3.1. Input Features

First, the input audio signal is filtered by a band-pass filter with low and high cut-off frequencies of 20 Hz and 4 kHz, respectively [9]. Then, the filtered signal is split into frames using a 30 ms Hann window with a 10 ms shift. Finally, 40 MFCCs are computed from each time frame and the resulting two-dimensional matrix is, after mean and standard deviation normalization, the input to the model [9].

In case of a multi-microphone signal (as the one from a hearing assistive device), we apply the above procedure independently to each channel and the resulting MFCC matrices are stacked across the quefrency dimension. While the number of parameters of the multi-task deep residual network remains the same, the number of multiplications experiences a relative increase of approximately $105 \times (N - 1)\%$, where N is the number of microphones.

3.2. Loss Weight Selection

To dynamically prioritize the most difficult task during training we experimented with dynamic task prioritization [12]. Let us denote the KWS and own-voice/external speaker detection tasks as T_1 and T_2, respectively, and let $\mathcal{T} = \{T_1, T_2\}$ designate the total set of tasks. Furthermore, let $\lambda_j(i)$ and $\mathcal{L}_j(i)$ denote the loss weight and training loss, respectively, for the j-th task at epoch i. The total loss at the i-th epoch can be expressed as

$$\mathcal{L}(i) = \sum_{j=1}^{|\mathcal{T}|} \lambda_j(i)\mathcal{L}_j(i), \qquad (1)$$

where $|\cdot|$ means cardinality. In [12], the loss weights are updated according to

$$\lambda_j(i) = -(1 - \bar{\kappa}_j(i)) \log(\bar{\kappa}_j(i)), \qquad (2)$$

where $\bar{\kappa}_j(i)$ is the training accuracy resulting from an exponential moving average with forgetting factor $\alpha = 0.75$. Then, the loss weights are normalized across tasks in such a way that $\sum_j \lambda_j(i) = |\mathcal{T}| \;\; \forall i$. Thus, the higher the training accuracy for a given task, the lower its loss weight.

We also tested two variants of dynamic task prioritization. The first one simply consisted of using $\lambda_j(i) = \bar{\kappa}_j^{-1}(i)$ instead of (2). Inspired by [13], the second variant considered the training loss instead of the training accuracy. Let $\bar{\mathcal{L}}_j(i)$ be the exponential moving average of $\mathcal{L}_j(i)$. The loss weights are updated with the loss ratio $\bar{\mathcal{L}}_j(i)/\mathcal{L}_j(0)$, where $\mathcal{L}_j(0) = \log(C_j)$ is a theoretical initial cross-entropy loss across C_j classes. This loss

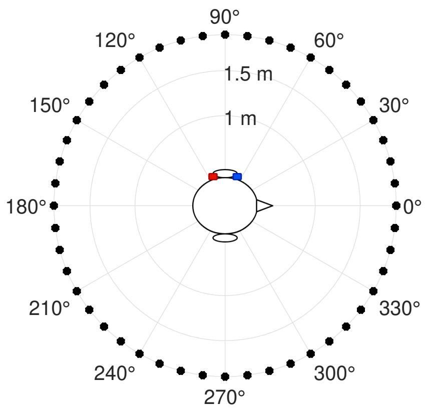

Figure 2: *Every external speaker can be located in one of the 48 equidistantly spaced points (black dots) on a circumference of 1.9 meter radius. An actual person wearing a 2-microphone behind-the-ear hearing aid in her left ear is seated in the center of the circumference. The blue and red dots symbolize the front and rear microphones, respectively, of the hearing aid.*

ratio can be understood as a measure of the inverse training rate of task T_j [13]. Finally, we again normalize the loss weights across tasks as described above. In this case, the lower the loss ratio for a given task, the lower its loss weight.

Our preliminary experiments showed that there are no statistically significant differences in terms of both KWS and own-voice/external speaker detection accuracies with respect to using constant equal loss weights, i.e., $\lambda_j(i) = 1 \;\; \forall i, j$. Therefore, as using constant equal loss weights is also the less computationally expensive approach, we will only present results using this scheme.

4. Experimental Framework

4.1. Hearing Aid Speech Database

The Google Speech Commands Dataset (GSCD) [10] is a speech database comprising 105,829 one second long utterances from a total of 2,618 different speakers. As each utterance contains only one word among a set of 35 possible words, this database is well suited for research on KWS. The GSCD also provides six different background noise files.

To create our speech database from the GSCD, we consider the scenario depicted in Figure 2. In a low-reverberation listening room, a circular array of 16 loudspeakers are placed equidistantly spaced around an actual sitting person with a diameter of 3.8 meters at eye-height. Then, 48 head-related transfer functions (HRTFs) are measured at an angular resolution of 7.5 degrees by rotating the chair on which person sits [14]. Here, an HRTF refers to the pair of acoustic transfer functions between the source (loudspeaker) and the front and rear microphones of the left ear hearing aid. Similarly, the own-voice transfer function (OVTF) from the mouth of the person to the microphones of her left ear hearing aid is also measured using a reference microphone placed 2 cm in front of the person's mouth. As measurements were recorded at a sampling rate of 44.1 kHz, the GSCD was upsampled prior to filtering the GSCD signals with the impulse responses.

Around 80% of the GSCD is reserved for model training, while the validation and test sets span another 10% each. Speakers do not overlap across sets. For each set, around 75% of the speakers are randomly selected to simulate that they wear hearing aids (own-voice subset). The rest of speakers (external speaker subset) are used to simulate external speakers and the external speaker angle with respect to the simulated user is randomly chosen from the set of 48 angles (see Figure 2) on an utterance basis.

For experimental purposes, we train our models to recognize 10 keywords: "yes", "no", "up", "down", "left", "right", "on", "off", "stop" and "go". The remaining 25 words of the GSCD are utilized to populate the *unknown word* class. This class, which is balanced across sets, represents around 10% of the utterances finally employed.

Table 1: *Summary of the distinguishing features of the different systems that are tested.*

	Architecture	**Training data**	**Input type**
Baseline	`res15` (KWS only)	Own voice	Front and rear mics
Front	Multi-task	Own and external voice	Front mic
Rear	Multi-task	Own and external voice	Rear mic
Dual	Multi-task	Own and external voice	Front and rear mics

4.2. Implementation and Training Issues

Data augmentation is applied during training on an utterance basis by taking into consideration the procedure outlined in [15]. First, a time shift of u ms is applied to the utterance, where u is drawn from the uniform distribution $\mathcal{U}(-100, 100)$. Next, with a probability of 0.8, a noise segment is randomly cut from one of the background noise files of the GSCD, scaled by a random factor between 0 and 1, and added to the time-shifted utterance. 30% of the training data is regenerated at each epoch [9].

The multi-task deep residual network was implemented using Keras [16]. Similarly to [9], different models were trained for a total of 26 epochs (which is more than enough for convergence) by stochastic gradient descent with a momentum of 0.9. Learning rate and learning rate decay were set to 0.1 and 10^{-5}, respectively. The minibatch size was of 64 training samples. For both KWS and own-voice/external speaker detection tasks, accuracy (i.e., the ratio between the number of correct predictions and the total number of predictions) was considered as performance metric.

5. Results

We test the multi-task architecture by making use of the dual-microphone signal (*Dual*) from the hearing assistive device and compare it with using the single-microphone signal from the front (*Front*) and rear (*Rear*) microphones, respectively. To assess the KWS performance of existing systems, which do not take the potential presence of external speakers into account, we test the original `res15` (i.e., with no own-voice/external speaker detection output) using the dual-microphone signal as input (Baseline). For Baseline model training, only own voice —and not external speaker data— is employed. For the sake of clarity, Table 1 summarizes the distinguishing features of the different systems that are tested.

5.1. Own-Voice/External Speaker Detection Results

The left part of Table 2 presents the own-voice/external speaker detection accuracy results[1], in percentages, with 95% confidence intervals across 10 different networks trained with dif-

[1]These results were obtained by making use of a sigmoid decision threshold of 0.5, which is equivalent to detecting the most likely class.

Table 2: *Own-voice/external speaker detection and KWS accuracy results, in percentages, with 95% confidence intervals.*

	Own-voice/External speaker detection			**Keyword spotting**	
	Own-voice subset	*External speaker subset*	*Overall*	*Own-voice subset*	*Overall*
Baseline	—	—	—	94.21 ± 0.39	71.87 ± 0.30
Front	97.49 ± 1.02	80.38 ± 5.23	93.02 ± 0.76	94.28 ± 0.37	89.48 ± 0.74
Rear	97.28 ± 1.08	79.03 ± 5.06	92.51 ± 0.68	94.48 ± 0.25	89.29 ± 0.55
Dual	99.60 ± 0.22	96.22 ± 1.61	98.72 ± 0.29	94.59 ± 0.32	94.86 ± 0.39

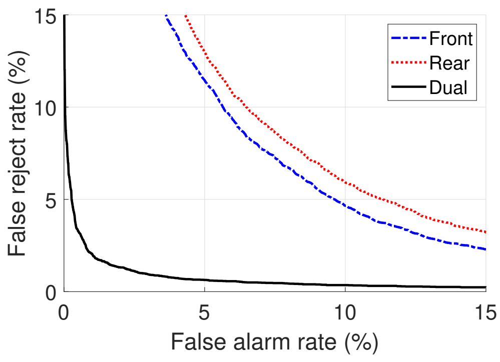

Figure 3: *Detection error trade-off curves for own-voice/external speaker detection.*

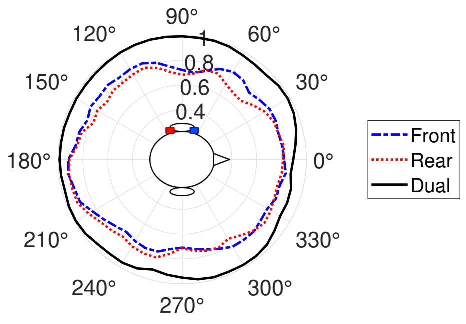

Figure 4: *Normalized external speaker detection accuracy as a function of the external speaker angle with respect to the user of the hearing assistive device. The users' head is centered in the origin and faces towards 0°.*

ferent random model initialization. Overall (i.e., over the whole test set) own-voice/external speaker detection accuracy results are also broken down by accuracies measured separately on the own-voice and external speaker subsets of the test set.

From the left part of Table 2, we can see that own-voice detection is accomplished with a high accuracy (between 97% and 99%) by exploiting either the front or rear microphone or both of them simultaneously. Using the dual-microphone signal helps for external speaker detection by clearly outperforming the single-channel strategy. These results are supported by the detection error trade-off curves for own-voice/external speaker detection in Figure 3. In this figure, pairs of false alarm rate and false reject rate values are plotted as a function of the sigmoid decision threshold (which is swept from 0 to 1). The smaller the area under the curve, the better a system is. Exploiting either the front or rear microphone works similarly with no statistically significant differences, and using the dual-microphone signal provides the best detection performance by far.

Figure 4 depicts the normalized external speaker detection accuracy as a function of the external speaker angle with respect to the hearing aid user. We hypothesize that the particular contour of these curves may be explained by the characteristics of the OVTF and HRTFs. This hypothesis is supported by an analysis of the OVTF and HRTFs, which showed that they are more similar (in terms of MFCC Euclidean distance) at angles where we see a relative drop in performance (e.g., for *Dual*, at frontal and shadow side [∼250°] angles). In other words, these similarities would make an external speaker less distinguishable from the user, thereby yielding a relative worsening in terms of external speaker detection accuracy.

5.2. Keyword Spotting Results

The right part of Table 2 presents the KWS accuracy results. For overall KWS accuracy computation, own-voice/external speaker detection is taken into account. Thus, correct decisions are made in the following two cases: *1)* when the hearing aid user triggers the system as a result of right keyword recognition, and *2)* otherwise (i.e., when the user speaks but it is not a keyword, or an external speaker utters a keyword or something different) the KWS system is not triggered. To understand the KWS performance degradation due to external speakers, KWS results on the own-voice subset of the test set are also reported.

The right part of Table 2 reveals the impact of own-voice/external speaker detection on KWS accuracy. Baseline results justify the need for effective own-voice/external speaker detection, as we propose. In other words, while the `res15` architecture achieves a high own-voice KWS accuracy, its performance drops significantly in the presence of external speakers. This performance degradation is clearly alleviated by using the multi-task architecture along with either the front or rear microphone signals. As expected, the best KWS accuracy results are obtained by jointly exploiting the multi-task learning scheme and the dual-microphone signal. This approach (∼94.86% acc.) achieves an overall KWS accuracy relative improvement of around 32% with respect to Baseline (∼71.87% acc.) and, more importantly, there is no drop between own-voice and overall KWS accuracies.

6. Conclusions

In this paper we have proposed a multi-task learning strategy to carry out KWS for hearing assistive devices that is robust to external speakers. This robustness is important for practical applications like the one assessed here, where our approach has been able to significantly outperform a state-of-the-art small-footprint KWS system. Furthermore, this has been achieved with a negligible increase in the number of parameters of the model.

7. Acknowledgements

This work was supported, in part, by the Oticon Foundation.

8. References

[1] M. B. Hoy, "Alexa, Siri, Cortana, and more: An introduction to voice assistants," *Medical Reference Services Quarterly*, vol. 37, pp. 81–88, 2018.

[2] S. O. Arik, M. Kliegl, R. Child, J. Hestness, A. Gibiansky, C. Fougner, R. Prenger, and A. Coates, "Convolutional recurrent neural networks for small-footprint keyword spotting," in *Proceedings of INTERSPEECH 2017 – 18th Annual Conference of the International Speech Communication Association, August 20-24, Stockholm, Sweden*, 2017, pp. 1606–1610.

[3] G. Chen, C. Parada, and T. N. Sainath, "Query-by-example keyword spotting using long short-term memory networks," in *Proceedings of ICASSP 2015 – 40th IEEE International Conference on Acoustics, Speech and Signal Processing, April 19-24, Brisbane, Australia*, 2015, pp. 5236–5240.

[4] T. J. Hazen, W. Shen, and C. White, "Query-by-example spoken term detection using phonetic posteriorgram templates," in *Proceedings of ASRU 2009 – IEEE Workshop on Automatic Speech Recognition & Understanding, December 13-17, Merano, Italy*, 2009, pp. 421–426.

[5] G. Chen, C. Parada, and G. Heigold, "Small-footprint keyword spotting using deep neural networks," in *Proceedings of ICASSP 2014 – 39th IEEE International Conference on Acoustics, Speech and Signal Processing, May 4-9, Florence, Italy*, 2014, pp. 4087–4091.

[6] R. Prabhavalkar, R. Alvarez, C. Parada, P. Nakkiran, and T. N. Sainath, "Automatic gain control and multi-style training for robust small-footprint keyword spotting with deep neural networks," in *Proceedings of ICASSP 2015 – 40th IEEE International Conference on Acoustics, Speech and Signal Processing, April 19-24, Brisbane, Australia*, 2015, pp. 4704–4708.

[7] T. N. Sainath and C. Parada, "Convolutional neural networks for small-footprint keyword spotting," in *Proceedings of INTERSPEECH 2015 – 16th Annual Conference of the International Speech Communication Association, September 6-10, Dresden, Germany*, 2015, pp. 1478–1482.

[8] R. Tang, W. Wang, Z. Tu, and J. Lin, "An experimental analysis of the power consumption of convolutional neural networks for keyword spotting," in *Proceedings of ICASSP 2018 – 43rd IEEE International Conference on Acoustics, Speech and Signal Processing, April 15-20, Calgary, Canada*, 2018, pp. 5479–5483.

[9] R. Tang and J. Lin, "Deep residual learning for small-footprint keyword spotting," in *Proceedings of ICASSP 2018 – 43rd IEEE International Conference on Acoustics, Speech and Signal Processing, April 15-20, Calgary, Canada*, 2018, pp. 5484–5488.

[10] P. Warden, "Speech Commands: A dataset for limited-vocabulary speech recognition," *arXiv:1804.03209v1*, 2018.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of CVPR 2016 – Conference on Computer Vision and Pattern Recognition, June 26-July 1, Las Vegas, USA*, 2016, pp. 770–778.

[12] M. Guo, A. Haque, D.-A. Huang, S. Yeung, and L. Fei-Fei, "Dynamic task prioritization for multitask learning," in *Proceedings of ECCV 2018 – European Conference on Computer Vision, September 8-14, Munich, Germany*, 2018, pp. 270–287.

[13] Z. Chen, V. Badrinarayanan, C.-Y. Lee, and A. Rabinovich, "GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks," *arXiv:1711.02257v4*, 2018.

[14] A. H. Moore, J. M. de Haan, M. S. Pedersen, P. A. Naylor, M. Brookes, and J. Jensen, "Personalized signal-independent beamforming for binaural hearing aids," *in review*.

[15] TensorFlow.org Tutorials, "Simple audio recognition," https://www.tensorflow.org/tutorials/sequences/audio_recognition.

[16] F. Chollet *et al.*, "Keras," https://keras.io, 2015.