SUPERVISED NOISE REDUCTION FOR MULTICHANNEL KEYWORD SPOTTING

Yiteng (Arden) Huang, Thad Hughes, Turaj Z. Shabestary, Taylor Applebaum

Google Inc., USA

{ardenhuang, thadh, turajs, applebaum}@google.com

ABSTRACT

This paper presents a robust, small-footprint, far-field keyword spotting (KWS) algorithm, which was inspired by the human auditory system's ability to achieve the so-called cocktail party effect in adverse acoustic environments. It introduces the idea of combining microphone-array speech enhancement with machine learning, by incorporating a feedback path from the neural network (NN) KWS classifier to its signal preprocessing frontend so that frontend noise reduction can benefit from, and in turn, better serve backend machine intelligence. We find that the new system can significantly improve KWS performance for Google Home when there is strong music or TV noise in the background. While this innovative and successfully validated strategy of combining signal processing and machine learning is developed for KWS, its technical feasibility is presumably extensible to many other applications, including noise robust speaker identification and automatic speech recognition.

Index Terms— Keyword spotting, supervised noise reduction, smart speech enhancement, microphone array processing for machine learning, cocktail-party effect

1. INTRODUCTION

Voice-based search has recently enjoyed unprecedented and still accelerating growth [1]. At Google, more than 20% of search queries are driven by speech. As voice-first devices (e.g., Google Home and Amazon Echo) are making their way into millions of homes, speech is becoming an essential way for people to search for assistance from today's rapidly changing digital world.

Speech interfaces ought to be completely hands-free, and it is a common practice to summon voice-based services by saying some preset keywords. Keyword spotting (KWS) is thus the entrance portal and a privacy safeguard[1] of many Google speech services, like Assistant on Google Home and Android phones. Its accuracy and robustness to noise and far-field distortions are crucial to consumer acceptance and loyalty.

Practical on-device KWS systems need to continuously listen to audio inputs from always-on microphones. Hence they should have a small memory and power footprint. Moreover, on-line KWS algorithms must have low latency. Early KWS approaches meeting these requirements were based on hidden Markov modeling (HMM) techniques [2, 3, 4]. Recent work on this topic followed great successes of using deep neural networks (DNNs) for automatic speech recognition (ASR) [5]. They were built on either recurrent neural network (RNN) [6, 7, 8], DNN [9], or convolutional neural network (CNN) [10] structures.

Our research on KWS algorithms at Google began with a DNN-based classifier [9], which was then replaced by a CNN-based system [10]. Multi-style training (MTR), automatic gain control (AGC), and a per-channel energy normalization (PCEN) frontend were developed to increase robustness to noise and far-field distortions [11, 12, 13]. Even after rounds of optimization and fine tuning, the performance of our state-of-the-art KWS methods still degraded significantly when the background has music or TV noise. In these circumstances, machine intelligence has not yet achieved the capability that human auditory systems demonstrate in the so-called cocktail party effect [14].

The cocktail party effect refers to our ability to selectively focus our hearing on one particular conversation, filtering out all other conversations and noise around us. Our auditory system does this by mediating both sensory and cognitive resources (ears/cochleae and brain, respectively). The mediation is never a unilateral service but rather involves both sensory-driven bottom-up and cognitive-directed top-down processing [15]. By analogy, prior KWS systems implemented only the bottom-up mechanism, without the top-down feedback. In this research we propose a new system architecture, feeding the neural network's outputs of keyword scores back into the multichannel frontend preprocessor such that background noise can be more effectively canceled. The new system enables our voice-based assistants to hear the user even in adverse acoustic environments.

2. SYSTEM OVERVIEW

As shown in Fig. 1, we feed keyword scores estimated by the CNN back into the microphone array preprocessor so that the preprocessor can better reduce noise for improved KWS. This system works more like the human auditory system, with both bottom-up and top-down processing between its sensory and cognitive subsystems. Each part of this new system will be discussed in more detail in the following sections.

3. CNN CLASSIFIER

Our KWS framework uses a CNN for acoustic modeling. The network consists of five layers, with three hidden layers between the input and output layers. A single-channel microphone signal is first segmented into 25 ms long frames with 10 ms hops. For each frame, 40-channel log-mel-filter-bank energies are computed and normalized using PCEN [13]. The input to the CNN consists of stacked feature vectors, with 24 left and 15 right context frames. On top of the input layer sits a convolutional layer which sweeps 308 non-overlapping 8×8 patches across time and frequency. That convolutional layer is followed by a linear projection layer with 32 outputs and a fully connected rectified linear unit (ReLU) layer with 128 outputs. The neurons in the output layer use a softmax activation function. There is one output (posterior probability) for each of the

[1]Google voice services only listen after the spotted keywords.

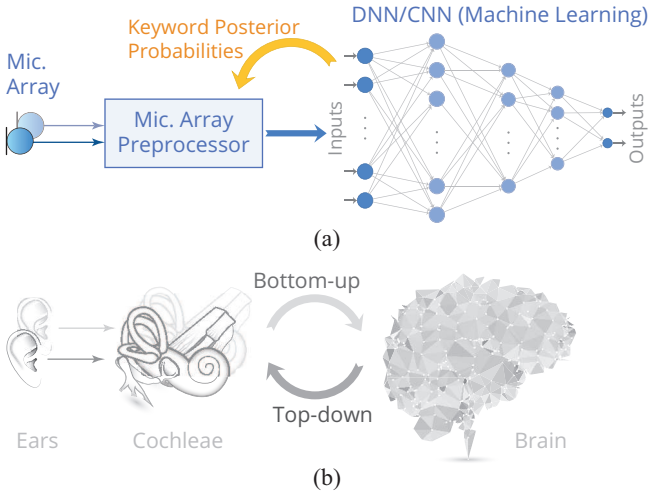

Fig. 1: Structure of the proposed KWS system (a) in comparison to that of human auditory systems (b).

phoneme targets in the keyword phrases ("Ok Google" and "Hey Google"), plus a single additional output representing all frames that belong to none of the aforementioned target classes. For more background information about our CNN structure for KWS, please read [10] and the references therein.

4. KEYWORD SIFTER: SUPERVISED SPEECH ENHANCEMENT

Our core KWS system operates on monophonic audio, but we would like to leverage stereo microphones to improve performance. To do this, we allow the KWS system to supervise a stereo speech enhancement algorithm.

As illustrated by Fig. 2, when a new frame of stereo audio samples is captured, the left channel (without loss of generality) is used by the monophonic CNN-based KWS classifier to calculate the keyword score of the new frame in its context. By comparing the score against two preset thresholds, the new frame is tagged with one of the three class labels: *trigger* (higher than the high threshold), *near-trigger* (between the two thresholds), and *noise* (lower than the low threshold). The high threshold is the threshold of the KWS classifier, while the low threshold is a small fraction of the high threshold. The new data frame and its class label are pushed into a first-in-first-out (FIFO) buffer of fixed length.

A controller is designed to deal with the data in the buffer. Depending on the frame tags, we face one of the following three situations:

a) If the data frames in the buffer are all *noise* frames, the controller outputs only one data frame from the buffer. Between the two channels of this popped frame, we apply adaptive noise cancellation (ANC) [16]: an ANC filter is convolved with the right channel and the result is subtracted from the left channel to form an error signal; the filter coefficients are then adjusted to minimize the mean square power of the error signal. Meanwhile, the error signal is output as a frame of the filtered utterance, which is sent to the second CNN for further processing.

b) When the buffer contains some *near-trigger* but no *trigger* frames, the whole buffer will be flushed. Since the chunk of data looks like a keyword but fails to trigger the first CNN, this actually is the moment in the utterance that we should double-check to avoid false rejects (FRs). The current ANC filter is used to process all the stereo audio samples in the buffer. But note that the filter coefficients are not updated in this process. The rationale behind this is that we want to suppress background noise without canceling keywords.

c) When the buffer contains at least one *trigger* frame, a keyword has been successfully identified by the first CNN. There is no need to over-clean the data. So we simply flush the buffer and pass the left channel of all the data frames in the buffer to the second CNN. As long as the buffer is long enough, the second CNN will make the same decision as the first CNN with respect to this chunk of data.

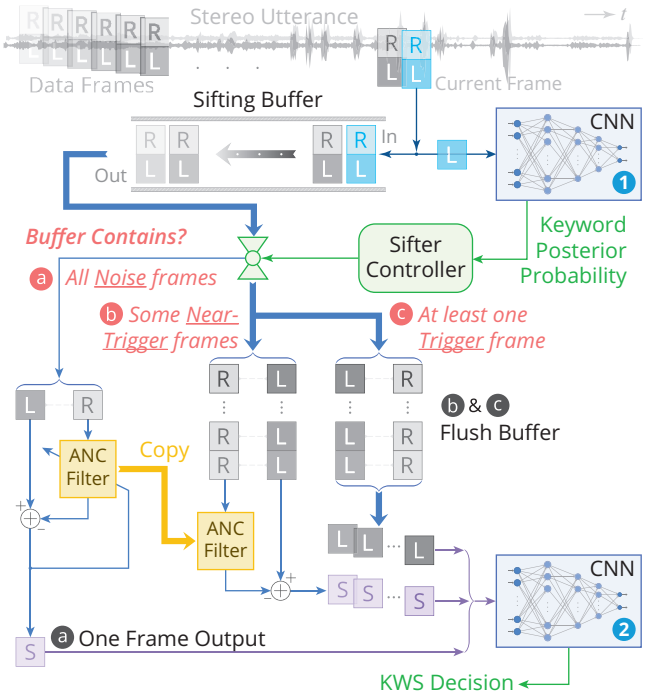

Fig. 2: Illustration of the keyword sifter. Note that ANC stands for adaptive noise cancellation.

We see that this keyword mining algorithm is in some ways similar to gold sifting. By analogy, when one sifts for gold, good-sized gold nuggets (*triggers*) are always taken directly out of the sifting pan first. Meanwhile, fine or granular sand (*noise*) that is easy to separate out is used to characterize all of the sand in the mixture. This learned knowledge about the background is helpful when sifting the rest of the mixture, which may contain either gold or sand (*near-trigger*). As a result, our proposed new system is named a keyword sifter and the buffer is called a sifting buffer.

An important point is the length of the sifting buffer. If it is too long, the adaptation of the ANC filter would have been halted much earlier before the leading edge of a keyword. This makes the ANC filter less effective to track background noise: the noise at the keyword becomes remarkably different from the noise used to estimate the ANC filter. Consequently only a low gain in noise reduction can be produced. On the other hand, if the buffer length is too short, the sifting buffer cannot accommodate a whole keyword instance. Then different phonemes of the keyword may be filtered in significantly different ways, yielding too much distortion and offsetting the large

Table 1: Summary of the STFT-domain fast RLS algorithm. Note that we have omitted $j\omega$ and denoted $\mathbf{P}(m) \triangleq \mathbf{R}^{-1}_{\mathbf{x}_2\mathbf{x}_2}(m)$ for clarity of presentation.

Parameters:
L = filter length, λ = forgetting factor
δ = coefficient to initialize $\mathbf{P}(0)$
Initialization:
$\mathbf{h}(0) = \mathbf{0}$, $\mathbf{x}_2(0) = \mathbf{0}$,
$\mathbf{P}(0) = \delta^{-1}\mathbf{I}$, where $\mathbf{I}$ is the identity matrix of rank L.
Adaptation: for $m = 1, 2, \cdots$
A-priori error:
$E(m) = X_1(m) - \mathbf{h}^H(m-1)\mathbf{x}_2(m)$,
Kalman gain vector:
$\mathbf{g}(m) = \dfrac{\mathbf{P}(m-1)\mathbf{x}_2(m)}{\lambda + \mathbf{x}_2^H(m)\mathbf{P}(m-1)\mathbf{x}_2(m)}$,
Update:
$\mathbf{P}(m) = \lambda^{-1}\left[\mathbf{P}(m-1) - \mathbf{g}(m)\mathbf{x}_2^H(m)\mathbf{P}(m-1)\right]$,
$\mathbf{h}(m) = \mathbf{h}(m-1) + \mathbf{g}(m)E^*(m)$.

gain in noise reduction. For "Ok Google" and "Hey Google" this length is empirically set to 1.5 s in our system.

Finally we would like to point out that while ANC is used here, other microphone array noise reduction techniques (e.g., beamforming [17]) can also be incorporated in this framework.

5. STFT-DOMAIN ADAPTIVE NOISE CANCELLATION

There are many widely-used adaptive algorithms that can be applied to implement the ANC subsystem [18]. Here we choose to present a short-time Fourier transform (STFT) based recursive least squares (RLS) method for its flexibility and computational efficiency when in combination with KWS feature extraction.

At frame m and for radian frequency ω, the STFT coefficients of the two microphone signals are denoted as $X_1(j\omega, m)$ and $X_2(j\omega, m)$, respectively, where $j \triangleq \sqrt{-1}$. The ANC filters the second microphone signal with a complex finite impulse response (FIR) filter of L taps

$$\mathbf{h}(j\omega, m) \triangleq \left[\, H_0(j\omega, m)\ H_1(j\omega, m)\ \cdots\ H_{L-1}(j\omega, m) \,\right]^T, \quad (1)$$

where $(\cdot)^T$ denotes the transpose of a vector or a matrix. The result is subtracted from the first microphone signal to get the error signal

$$\mathcal{E}(j\omega, m) \triangleq X_1(j\omega, m) - \mathbf{h}^H(j\omega, m)\mathbf{x}_2(j\omega, m), \quad (2)$$

where $(\cdot)^H$ is the Hermitian transpose of a vector or a matrix and

$$\mathbf{x}_2(j\omega, m) \triangleq \left[\, X_2(j\omega, m)\ X_2(j\omega, m-1)\ \cdots\ X_2(j\omega, m-L+1) \,\right]^T,$$

is the collection of the STFT coefficients of the second channel in the current and the last $L-1$ frames. The weighted least squares cost function is expressed as

$$J\left\{\mathbf{h}(j\omega, m)\right\} \triangleq \sum_{i=0}^{m} \lambda^{m-i} \left|\mathcal{E}(j\omega, i)\right|^2, \quad (3)$$

where $0 < \lambda \leq 1$ is the forgetting factor. This cost function is minimized by taking the partial derivatives of (3) w.r.t. $\mathbf{h}^H(j\omega, m)$ and setting the result to zero

$$\frac{\partial J}{\partial \mathbf{h}^H(j\omega, m)} = \mathbf{R}_{\mathbf{x}_2\mathbf{x}_2}(j\omega, m)\mathbf{h}(j\omega, m) - \mathbf{r}_{\mathbf{x}_2 x_1}(j\omega, m) = \mathbf{0}, \quad (4)$$

where

$$\mathbf{R}_{\mathbf{x}_2\mathbf{x}_2}(j\omega, m) \triangleq \sum_{i=0}^{m} \lambda^{m-i}\mathbf{x}_2(j\omega, i)\mathbf{x}_2^H(j\omega, i),$$
$$\mathbf{r}_{\mathbf{x}_2 x_1}(j\omega, m) \triangleq \sum_{i=0}^{m} \lambda^{m-i}\mathbf{x}_2(j\omega, i)X_1^*(j\omega, i),$$

and $(\cdot)^*$ is the conjugate of a complex variable. Solving (4) for $\mathbf{h}(j\omega, m)$ yields the RLS solution

$$\mathbf{h}_{\text{RLS}}(j\omega, m) = \mathbf{R}^{-1}_{\mathbf{x}_2\mathbf{x}_2}(j\omega, m)\mathbf{r}_{\mathbf{x}_2 x_1}(j\omega, m). \quad (5)$$

Using the Woodbury's identity [19], direct computation of the matrix inversion in (5) can be avoided and a computationally more efficient version is deduced. For brevity, this fast RLS algorithm is summarized in Table 1.

6. EXPERIMENTS

6.1. Experimental Setup

Our keyword sifter system is based on the mono-channel CNN classifier as described in Sec. 3. The CNN was implemented using Google's TensorFlow™ library [20] and was trained with deep learning algorithms on large internal training data collected from a gender-balanced pool of volunteers with a variety of accents from around the world [21]. The acoustic feature extraction module and the CNN runtime engine are both implemented using fixed-point arithmetic in order to minimize power consumption [22].

The sifter works on a Google Home [23] device with two microphones separated by a distance of 71 mm. Its performance is compared against two independent CNN baseline systems that take the two microphone signals as their inputs. It is worth noting that these baseline systems and the two KWS subsystems of the sifter are exactly the same.

In this study, we used re-recorded data on a Google Home in a 10 m by 8 m living-room lab with a couch, chairs, tables, a coffee table, a TV, and hardwood floors. Speech utterances with either "Ok Google" or "Hey Google" were collected from 50 volunteers (a gender-balanced pool of adults and some kids) and played back from a mouth simulator at five different radial positions relative to the Google Home at a distance ranging from 1.5 m to 5 m. The background was either quiet or filled with music or TV noise. Music was streamed from Google Play Music and played with two KRK Rokit external loudspeakers. TV contents were pulled from YouTube including movies, cartoons, and news. Each of the re-recorded utterances has a known signal-to-noise radio (SNR) of 0-10 dB. We have also re-recorded a much larger set of TV noise which do not contain any of the target keywords and are hence used as the 'negative' evaluation dataset. In this recording, the Google Home was placed at three different distances (0.5, 1.5, and 2.5 m) from the TV, each providing about the same length of audio. Table 2 presents a summary of the evaluation data. These data have no overlap with those used for training our CNN KWS system and all of them are from volunteers.

Re-recorded data did not take into account of the Lombard effect [24] which refers to the involuntary tendency of speakers to increase their vocal effort (loudness, pitch, rate, duration of syllables, etc.)

Table 2: Summary of collected data for performance evaluation.

Dataset	Number of Utterances	Length (hours)
Clean	1831	6.23
Music	1520	5.83
TV	1455	5.51
Negative	7528	229.32

when speaking in loud noise to enhance the audibility of their voice. So the KWS performance degradation measured on re-recorded data may not reflect the true loss due to noise. But this is a fast and economic way to collect a large amount of speech utterances in actual rooms at controlled SNRs, and nevertheless the re-recorded data is valid for justifying the proposed sifter in comparison with the baseline systems.

We measure the performance of a KWS system using the well-known receiver operating curve (ROC), which visualizes the false reject (FR) rate (in number of FRs per instance) as a function of the false accept (FA) rate (in number of FAs per hour). The lower the FR at a preset FA rate, the more accurate is the KWS system. Practically we aim to reduce the FR rate while keeping the FA rate at a very low level to maximize user satisfaction.

6.2. Results

Let's first examine how the sifter works on far-field speech utterances in a quiet acoustic condition. Figure 3(a) presents the evaluation results. All of the three KWS systems are highly accurate on this clean dataset. There are some disparities between the two baseline ROCs at low FA rates but the difference is insignificant at the operating point of 0.02 FA/hr. So a simple logical-or combination of these two systems won't be able to offer much gain in accuracy. The sifter system doesn't degrade the performance, which is noteworthy.

The comparisons in ROC among the three KWS systems on speech utterances with music and TV noise are plotted in Fig. 3(b) and (c), respectively. These are two very challenging evaluation datasets for KWS, containing a large number of utterances of low SNRs. In these utterances, keywords get submerged in background noise and even human listeners can sometimes have difficulties to spot them. Moreover, since the last phoneme of "Ok Google" and "Hey Google" is a consonant, it is not an uncommon pronunciation in which the sound fades out at the tail. So the last phoneme target of the two keywords tends to be easily masked by noise in noisy environments with a much lower than average SNR. This inherent feature of our unique keyword phrases makes the KWS task more problematic, leading to high FR rates. The sifter system evidently outperforms the baseline systems, with relative average improvements of 32.0% (music) and 35.9% (TV) in FR rates at the operating point. Informal listening tests of some of the sifted utterances also report that the sifter can effectively suppress the noise, making keywords emerge prominently from the background. This is consistent with what is revealed by a closer examination of target-level recognition of our keywords – e.g., the confidence scores of the last consonant are remarkably improved.

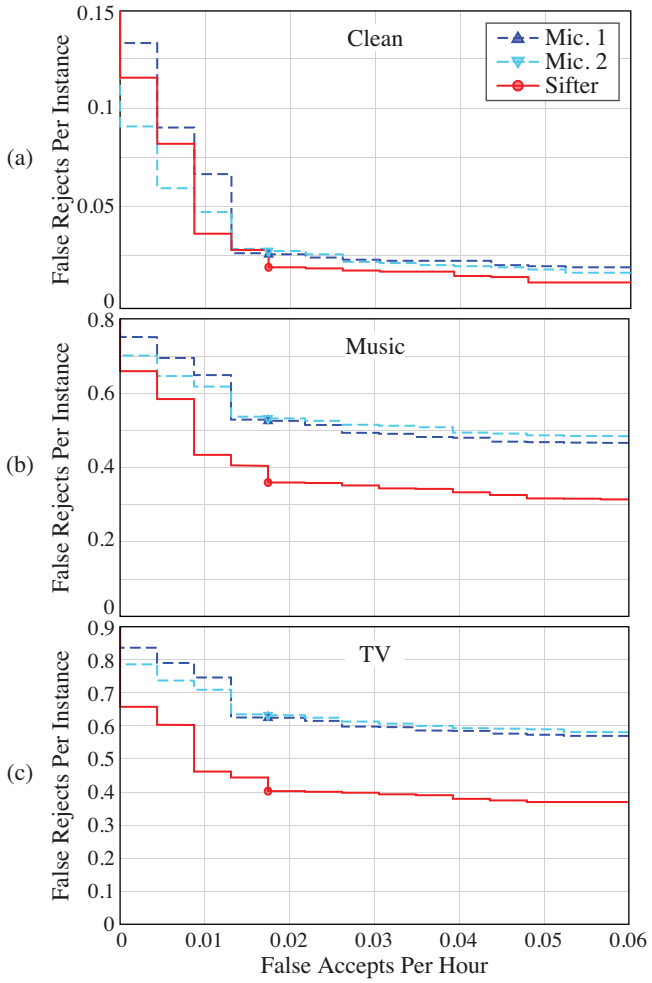

Fig. 3: ROC curves comparing performance of the keyword sifter against two baseline systems that use only one microphone signal of the keyword sifter system for (a) clean, (b) music (0-10 dB SNR), and (c) TV noise (0-10 dB SNR) evaluation datasets.

7. CONCLUSIONS

In this paper, we have presented a machine-supervised noise reduction framework for multichannel KWS. We explained that human auditory systems could accomplish the cocktail party effect thanks to the collaboration between both sensory-driven bottom-up and cognitive-directed top-down processing; but, by analogy, prior KWS methods followed only the bottom-up mechanism and missed out on the top-down path. A novel multichannel noise reduction algorithm was developed under the supervision of KWS machine intelligence. Feedback from the KWS neural network made the noise reduction algorithm more aware of its specific task such that it can exploit the knowledge estimated during non-keyword periods about surrounding noise and better filter out noise when keywords possibly appear. The KWS system based on this new algorithm worked more like human auditory systems. Using re-recorded speech utterances on Google Home, we found that the proposed framework produced a greater than 32% relative improvement in performance over the baseline systems in typical noisy conditions.

8. ACKNOWLEDGMENTS

The authors would like to thank Alex Gruenstein, Aleks Kracun, and Johan Schalkwyk for their invaluable technical support and constructive discussion of the ideas during this research.

9. REFERENCES

[1] J. Schalkwyk, D. Beeferman, F. Beaufays, B. Byrne, and C. Chelba, "'Your word is my command': Google search by voice: A case study," in *Advances in Speech Recognition*, A. Neustein, Ed., chapter 4, pp. 61–90. Springer, Boston, MA, 2010.

[2] R. C. Rose and D. B. Paul, "A hidden Markov model based keyword recognition system," in *Proc. IEEE ICASSP*, 1990, pp. 129–132.

[3] J. R. Rohlicek, W. Russell, S. Roukos, and H. Gish, "Continuous hidden Markov modeling for speaker-independent wordspotting," in *Proc. IEEE ICASSP*, 1990, pp. 627–630.

[4] J. G. Wilpon, L. G. Miller, and P. Modi, "Improvements and applications for key word recognition using hidden Markov modeling techniques," in *Proc. IEEE ICASSP*, 1991, pp. 309–312.

[5] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, September 2012.

[6] S. Fernández, A. Graves, and J. Schmidhuber, "An application of recurrent neural networks to discriminative keyword spotting," in *Proc. Int. Conf. Artificial Neural Networks*, 2007, pp. 220–229.

[7] M. Wöllmer, B. Schuller, and G. Rigoll, "Keyword spotting exploiting long short-term memory," *Speech Commun.*, vol. 55, pp. 252–265, February 2013.

[8] P. Baljekar, J. F. Lehman, and R. Singh, "Online word-spotting in continuous speech with recurrent neural networks," in *Proc. Spoken Language Technology Workshop (SLT)*, 2014, pp. 536–541.

[9] G. Chen, C. Parada, and G. Heigold, "Small-footprint keyword spotting using deep neural networks," in *Proc. IEEE ICASSP*, 2014, pp. 4087–4091.

[10] T. N. Sainath and C. Parada, "Convolutional neural networks for small-footprint keyword spotting," in *Proc. Interspeech*, 2015, pp. 1478–1482.

[11] C. Kim, A. Misra, K. Chin, T. Hughes, A. Narayanan, T. Sainath, and M. Bacchiani, "Generation of large-scale simulated utterances in virtual rooms to train deep-neural networks for far-field speech recognition in Google Home," in *Proc. Interspeech*, 2017, pp. 379–383.

[12] 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 *Proc. IEEE ICASSP*, 2015, pp. 4704–4708.

[13] Y. Wang, P. Getreuer, T. Hughes, R. F. Lyon, and R. A. Saurous, "Trainable frontend for robust and far-field keyword spotting," in *Proc. IEEE ICASSP*, 2017, pp. 5670–5674.

[14] E. C. Cherry, "Some experiments on the recognition of speech, with one and with two ears," *J. Acoust. Soc. Am.*, vol. 25, pp. 975–979, Sept. 1953.

[15] J. Driver, "A selective review of selective attention research from the past century," *British J. Psychology*, vol. 92, pp. 53–78, Feb. 2001.

[16] B. Widrow, J. R. Glover, J. M. McCool, J. Kaunitz, C. S. Williams, R. H. Hearn, J. R. Zeidler, E. Dong, and R. C. Goodlin, "Adaptive noise cancelling: principles and applications," *Proc. IEEE*, vol. 63, pp. 1692–1716, Dec. 1975.

[17] B. Van Veen and K. Buckley, "Beamforming: A versatile approach to spatial filtering," *IEEE ASSP Mag.*, vol. 5, no. 2, pp. 4–24, Apr. 1998.

[18] S. Haykin, *Adaptive Filter Theory*, Prentice-Hall, Upper Saddle River, NJ, 4 edition, 2002.

[19] M. A. Woodbury, "Inverting modified matrices," Tech. Rep. Memorandom Rept. 42, Princeton University, 1950.

[20] Google Inc., "TensorFlow™: An open-source library for Machine Intelligence," https://www.tensorflow.org, [Online; Latest Accessed 20-Oct-2017].

[21] T. Hughes, K. Nakajima, L. Ha, A. Vasu, P. Moreno, and M. LeBeau, "Building transcribed speech corpora quickly and cheaply for many languages," in *Proc. Interspeech*, 2010, pp. 1914–1917.

[22] X. Lei, A. Senior, A. Gruenstein, and J. Sorensen, "Accurate and compact large vocabulary speech recognition on mobile devices," in *Proc. Interspeech*, 2013, pp. 662–665.

[23] Google Inc., "Google Home - Smart Speaker & Home Assistant - Google Store," https://store.google.com/product/google_home, [Online; Latest Accessed 20-Oct-2017].

[24] H Lane and B. Tranel, "The Lombard sign and the role of hearing in speech," *J. Speech, Language, Hearing Res.*, vol. 14, pp. 677–709, Dec. 1971.