EVENT-DRIVEN PIPELINE FOR LOW-LATENCY LOW-COMPUTE KEYWORD SPOTTING AND SPEAKER VERIFICATION SYSTEM

Enea Ceolini, Jithendar Anumula, Stefan Braun and Shih-Chii Liu

Institute of Neuroinformatics, University of Zurich and ETH Zurich, Zurich, Switzerland

enea.ceolini@ini.uzh.ch,shih@ini.ethz.ch

ABSTRACT

This work presents an event-driven acoustic sensor processing pipeline to power a low-resource voice-activated smart assistant. The pipeline includes four major steps; namely localization, source separation, keyword spotting (KWS) and speaker verification (SV). The pipeline is driven by a front-end binaural spiking silicon cochlea sensor. The timing information carried by the output spikes of the cochlea provide spatial cues for localization and source separation. Spike features are generated with low latencies from the separated source spikes and are used by both KWS and SV which rely on state-of-the-art deep recurrent neural network architectures with a small memory footprint. Evaluation on a self-recorded event dataset based on TIDIGITS shows accuracies of over 93% and 88% on KWS and SV respectively, with minimum system latency of $5\,\mathrm{ms}$ on a limited resource device.

***Index Terms*—** silicon cochlea spikes, event-driven auditory processing, DNN, keyword spotting, speaker verification

1. INTRODUCTION

The demand for personalized voice-activated devices has rapidly grown in recent years. Along with this, we see increasing research in algorithms useful for these devices such as speaker verification (SV), and keyword spotting (KWS) [1, 2, 3].

Given that portable devices have limited memory and computational resources, algorithms that are cheap to compute and have a low memory footprint are preferred. For this reason, much effort has been focused on the use of small models that can be efficiently implemented on these devices [4, 5, 6, 7].

Two other major considerations for voice-activated devices are first, the ability to operate robustly in challenging noisy and multi-talker environments; and second, to provide low-latency responses to the user [8, 9, 10].

These requirements fit well with the advantages that can be offered by an event-based front-end low-power silicon audio sensor such as the Dynamic Audio Sensor (DAS) [11, 12] which implements an abstract model of the biological cochlea and outputs asynchronous and precisely timed events ($< 1\mu$s) at low latencies [13]. The DAS output spike streams have been used to drive low latency localization solutions together with the separation of the spike streams produced by competing talkers [14, 15]. These previous studies show that it is possible to simultaneously localize multiple speakers, separate their spike streams, and estimate the speech envelope of an individual speaker from the separated spike streams. The DAS cochlea has also been used in ASR tasks such as speaker identification [16], speech recognition [17], and voice activity detection [18].

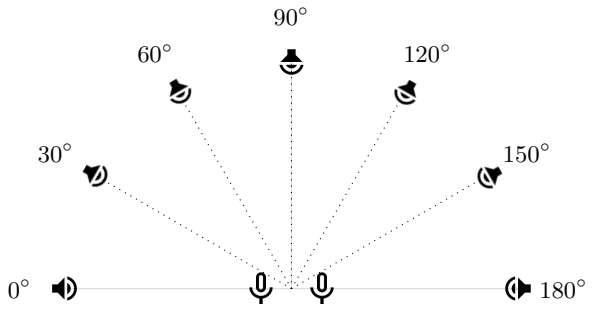

Fig. 1. Experimental setup of the recordings. The microphones are on the cochlea system. Speakers are played from one of the 7 loudspeakers.

This work presents a full pipeline for use in a voice activated smart assistant device with a front-end spiking cochlea. The pipeline consists of four sequential steps, namely, localization of simultaneously active speakers, speech separation, KWS and SV. This paper will demonstrate how the sparsity of the DAS events and the resulting features allow for models with a low memory footprint, low computational complexity and low-latency system response, thus providing a compelling alternative to standard solutions. The paper is structured as follows: Section 2 describes the methods and models, Section 3 describes related work, Section 4 presents results of the pipeline on a particular dataset and performance of the models when deployed on a limited resource device. Section 5 concludes the work.

2. METHODS

2.1. Recordings and dataset

The front-end sensor is the binaural spiking silicon cochlea or Dynamic Audio Sensor (DAS) system [11]. It has two independent 64-stage cascaded filter banks driven by two microphones. Each cascaded filter bank models the basilar membrane, inner hair cells, and spiral ganglion cells of the biological cochlea. The sensor details are described in [11, 12]. The frequency selectivity of the cochlea channels ranges from $100\,\mathrm{Hz}$ to $10\,\mathrm{kHz}$.

The DAS system [11] can be driven by the on-board microphones or from the computer through the on-board audio jacks. Instead of doing recordings in a specific room, in this work, we obtain microphone recordings by simulating the room impulse response (RIR) between each source and each microphone calculated based on the image method [19]. The microphone recordings from the simulated environment are then played to the DAS from the com-

puter. The simulated room has dimensions of $5 \times 5 \times 5\,\mathrm{m}^3$ and no reverberation (i.e. $T_{60} = 0$). The microphones which are spaced $20\,\mathrm{cm}$ apart, are placed in the center of the room. Even though a 3D environment is simulated, this work only considers azimuthal localization using speakers that are distributed in a semicircle with a 2m radius and are separated by 30° (see Fig. 1).

The recordings are based on the full TIDIGITS dataset [20] which consists of a series of spoken digit sequences and single digits. Each sample in the recordings consists of a mixture of the speech from two speakers in different positions. The mixtures are created by randomly selecting two speakers for each sample, and then using one random utterance for each chosen speaker. The list of utterances together with the spiking dataset is available for download [21]. Note that the original waveforms cannot be provided but the dataset is fully reproducible with the list of utterances. The code for creating the mixtures and the random impulse responses is also available.

The recordings include 6000 samples for training and 2000 for testing with 225 different speakers. The speakers are divided equally between males and females. The average number of sentences per speaker is 70 while the minimum and maximum sentences are 22 and 132 respectively. the framework provides the possibility for increasing the dataset size to fit any task requirement.

2.2. ITDs and probabilistic model

This work builds on top of the work presented in [14], where the authors used a binaural cochlea to localize concurrent speakers and showed how each spike can be assigned a probability of being produced by one of the speakers. Their approach uses the interaural time difference (ITD) between cochlea events from the two microphones (ears) in order to localize a sound source. If a sound triggers an event in one microphone, it will trigger a similar event in the second microphone with a delay that depends on the position of the source with respect to the microphones.

In short, given the kth event e_k at time t_k, from frequency channel c_k and at the ear r_k: $e_k = [t_k, c_k, r_k]$, the ITD is estimated by computing the time difference between e_k and the closest event from the opposite ear in the corresponding (same) frequency channel c_k. Note that only windows of maximal ITDs of $600us$ are considered, given by the $20cm$ distance between the two microphones. In previous work, only a limited subset of frequency channels in [14] are considered for localization. In the next subsection, we present a method that allows us to use more of the frequency channels.

The extracted ITDs drive a probabilistic Bayesian model that is used to track the positions of the speakers. A Hidden Markov Model is used to estimate the posterior probability of the position of the speaker given an ITD value as an observation. This algorithm is also known as Bayes Filter or Recursive Bayes Estimation.

Once probability distributions have been estimated for each spike, the maximum a posteriori (MAP) criterion is used to assign each spike to one of the sources.

2.3. Channel-wise delay correction

In [14], a considerable amount of spikes were discarded because of the underlying discrepancies of extracted ITDs created by the known mismatch of analog DAS circuits produced from the silicon fabrication process.

The authors only used the few cochlea channels that produce the expected ITD values without any calibration on the DAS outputs. A low number of spikes attributed to each speaker is not a problem

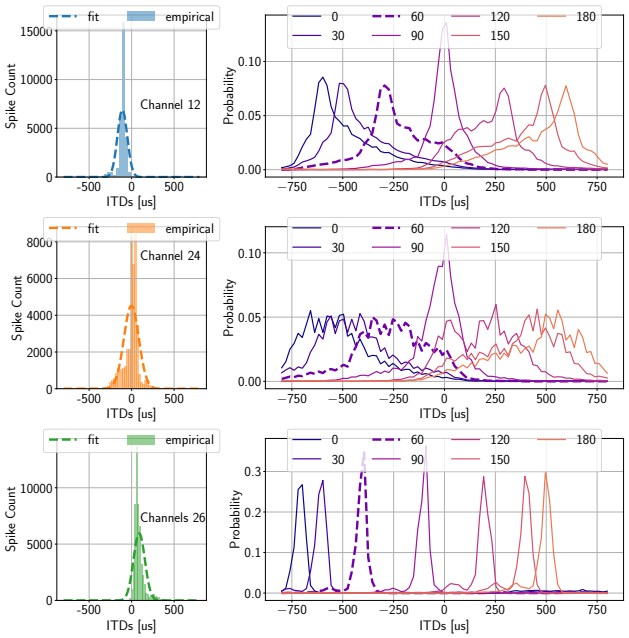

Fig. 2. Algorithm for ITD distribution correction for different cochlea channels. Left, example of ITD distributions and fitted Gaussian curves for a sound source at 90°. Note that the peak is shifted differently for each channel. Right, prior distribution in 3 cases: (top) after correction, (center) before correction, (bottom) before correction only if channel 19 is considered.

for envelope reconstruction, (e.g. in [14]), but it is a concern when trying to use spike features for more complex tasks such as speech recognition. For this reason, we use a delay correction algorithm here so we can use more cochlea channels for calculating the ITDs and then assigning these spikes to the probable locations.

Figure 2 shows the ITD histograms for a subset of cochlea channels responding to a stimulus positioned at 90° corresponding to zero delay between the two ears. Clearly, the ITD distributions for different channels have peak values different from zero. As shown in Fig. 2 (left column), an ITD histogram is created for each channel corresponding to a stimulus at 90°, then a Gaussian fit to this empirical distribution provides an estimated mean. This estimated mean shift can thus be applied to correct the ITD for any new sample. The right column of Fig. 2 shows how the priors change after the correction. The bottom panel shows the non-corrected ITD for channel 19, the central panel shows the non-corrected ITD for all channels while the top panel shows the combined ITD from all channels after correction. The number of spikes that can be used for assignment after the correction step, increases by a factor of 20.

2.4. Spike separation and spike features

In the scenario provided by the recorded dataset, multiple speakers can be active at the same time. The first step of the pipeline is to assign each spike to one of the speakers. As shown in [14, 15], one can use the output of the probabilistic model to assign to each spike, the probability of having being generated by one of the speakers in the possible positions. When a MAP criterion is applied to these posterior probabilities, it is possible to retrieve most of the spikes generated by each speaker. The separated spikes can be used, e.g.,

for reconstructing the power envelope of the speech of a speaker. In this work, the separated spike streams will be used as input to a system for SV and KWS.

The spike features (shown in Fig. 3, top left panel) are a key part of the pipeline. They are easy to compute and need to carry useful information to complete all the tasks in the pipeline. The features used in this work are spike counts with a window of $5\,\mathrm{ms}$ [22] followed by a logarithmic compression. These features are similar to log-filterbank features commonly used for state-of-art speech recognition systems.

2.5. Speaker verification

Customization is an important part of every smart system that is deployed in the real world. For smart assistants, SV is crucial in scenarios where multiple users share the same assistant. While the use of voice as a security measure is still under debate, it can be used by the smart assistant to redirect queries to different accounts based on different users' settings.

In this scenario, a SV system has to be able to work with a few voice samples from the users. Therefore, we use the spike features within a one-shot learning framework for this task. One successful method used in one-shot learning tasks employs a neural network architecture known as a Siamese network [23]. The network is trained to distinguish between samples from the same class or from different classes. This training is done by feeding both samples through a common part of the network and then letting the final layer decide if the encoded inputs are from the same class or from different classes. The common part of the network in this work is implemented by 2 stacked gated recurrent units (GRU) [24] layers with 220 units per layer and a fully-connected layer with 128 units and sigmoidal activation. Dropout with 0.3 probability is applied between the two GRU layers. Finally the L1 difference between the two encoded samples is used to drive a fully-connected layer with a single output unit and a sigmoidal activation. The role of this final unit is to encode the probability that the two samples belong to the same class. Note that this network has only $900\,\mathrm{k}$ parameters, a memory footprint of $3\,\mathrm{MB}$. Because of this, only a few milliseconds are needed to compute all layers during inference and can be run in real time because only unidirectional GRU layers are used.

2.6. Keyword spotting

Smart assistants are usually activated by a wake word [4]. To ensure that the wake word is always spotted, the system needs to constantly run a detection algorithm. For this reason, low-power and low-compute cost solutions to this problem are needed especially for portable devices. This work aims to use the separated spike streams to address this problem. The spike streams are ideal for this task because they are generated with low latencies; and in addition, they are sparse and asynchronous.

For this task, the keyword consists of a digit between 0 and 9. The sequence is marked with a flag every time the wake word is present. The model has to detect if the keyword appears in the sentence. The model used for this task is a recurrent neural network consisting of 3 stacked GRU layers. The network is trained with connectionist temporal classification (CTC) [25] to output either a blank symbol or the presence or absence of the wake word. The model has roughly $900\,\mathrm{k}$ parameters and a memory footprint of less then $3\,\mathrm{MB}$. Similar to the SV network, it uses only unidirectional layers and can thus be used in real time.

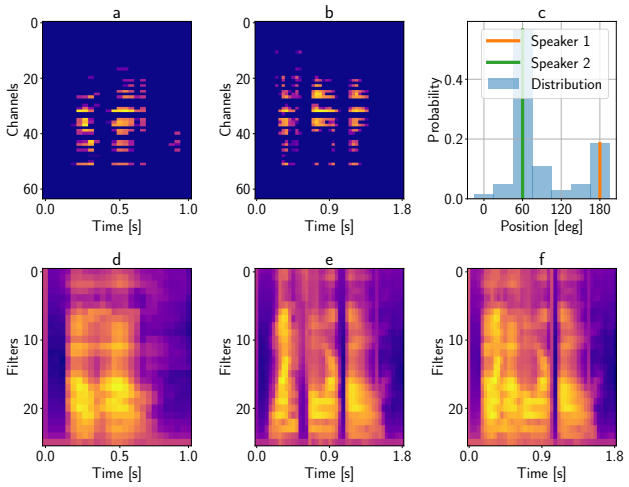

Fig. 3. Example of reconstructed spike features after the localization; and subsequent steps of separation and spike assignment. (a) and (b) show respectively the two separated spike streams from each speaker, utterances are '00' and '140'. (c) probability distribution for localization; the two local maxima indicate the positions of the two speakers. (d), (e) and (f) log-filterbank features of speaker 1, speaker 2, and the mixture respectively.

3. RELATED WORK

This work builds on top of [14, 15]. It uses the same localization and spike stream separation methods. Nevertheless, this work introduces a new method to increase the number of spikes assigned to each stream which is key for SV and KWS. Prior work where cochlea spikes are also used for a speaker identification task [16], differs in that it uses a support vector machine (SVM) classifier and a small dataset. It did not include testing on unknown speakers which we address here. Prior work also reported a speech recognition task using a recorded spike-based TIDIGITS dataset [22]. Here, we address the problem of keyword spotting in a more challenging setting with competing talkers. Recent studies tackle KWS for both efficient computation [4] and small footprint [6, 7], while not many studies address these problems in the context of SV [3, 2]. Efforts exist for either jointly solving both tasks [5] or solving a single task in the presence of background noise [10]. In contrast, this work proposes a solution that jointly solves both tasks in the harder scenario of competing talkers. It also leads to a low-latency response because of the use of event-based features.

4. EXPERIMENTAL RESULTS

4.1. Localization and spike separation

The probabilistic approach in [14] is used to estimate the locations of the speakers. The spikes assigned to those locations represent the speech of the corresponding speaker. Figure 3 shows an example of this part of the pipeline. Panel c shows the two local maxima in the distribution indicating the true location of the speakers, while panels a and b show the separated spike stream features. Notice how these features vaguely resemble the corresponding log-filterbank features calculated on the ground truth separate audio waveforms. The localization algorithm finds the correct speaker positions in 85% of the

cases. Of the remaining 15%, 10% belong to the cases where the position is off by 30°. In the remaining cases, the streams do not correlate well with the speech of the target speaker.

4.2. Speaker verification

The Siamese network described in Section 2.5 was trained on 200 speakers and tested on 25 unknown speakers. The results are summarized below and the standard deviations are obtained by training the model on different datasets where speakers for train and test are shuffled. Results are given in terms of equal error rate (EER) and accuracy in a 2-way one-shot learning task. The one-shot learning task consists of pairs of one positive and one negative example in which the outcome is successful if the probability for the positive sample is higher then the probability for the negative one.

The presented model achieves $78.5 \pm 2.3\%$ accuracy on a 2-way one-shot learning task, while typical state-of-the-art results are around 90% [5]. This is due to both the limited size of the network and the limited size of the dataset. Fortunately, the accuracy can be increased by fine tuning the network, that is, retraining the network with a small number of samples from the new speakers. The model achieves an accuracy of 80.3 ± 2.5, 84.6 ± 1.9 and 88.2 ± 1.5 after fine tuning with 1, 3 and 5 samples respectively from the new speakers. Equivalently, the EER decreases from 23.2 ± 2.1 to 13.5 ± 1.9 when fine tuning the network with 5 samples from the new speakers. The results can be improved by using a dataset with more speakers.

4.3. Keyword spotting

The model receives as input a sequence of spoken digits and should output the presence or absence of the keyword. Note that this is not a simple binary classification task since the model has to detect both the presence and the position of the word in the sequence. Although CTC does not provide precise alignment of words, it gives enough temporal precision to activate the system with little latency. Tests were carried for the 10 different keywords in the dataset.

The results are reported in terms of the $F_{0.5}$ scores which account more for loss in precision then recall. Ideally, such a system should avoid false positives as these activate the more costly parts. The results, as reported in Table 1, show clearly the statistical differences between $F_{0.5}$ and precision for different keywords. Note that only the best digits were reported. This is due to the spectral composition of the phonemes of the digits, therefore, some digits are easier to recognize then others. These results contrast with the ones obtained with more standard log-filterbank features which in this task obtain an accuracy of around 99%. They are also in contrast with the results obtained on the same task using spike features from a single speaker scenario which obtains an accuracy of roughly 97%. This loss in accuracy is due to the separation step of the pipeline that reduces conspicuously the number of spikes used by the recognition model. The loss of spikes reduces the quality of the spike features, thus the decrease in accuracy. In particular, the separation step of the pipeline reduces by 80% the number of spikes that are used to create the spike features with respect to the spikes produced by the sensor.

4.4. Resource requirements

It is important to note that the recurrent models we used are unidirectional and thus they allow for real-time computation. Moreover, both models have very small number of parameters therefore needing a low memory footprint and allowing fast inference. In addition, the spike features allow for low latency system responses.

Table 1. KWS results. Precision, recall and $F_{0.5}$ scores with mean and standard deviation obtained from 3 different initializations.

Keyword	Precision	Recall	$F_{0.5}$
1	0.88 ± 0.01	0.92 ± 0.05	0.89 ± 0.02
2	0.89 ± 0.01	0.68 ± 0.02	0.83 ± 0.05
4	0.93 ± 0.02	0.78 ± 0.01	0.90 ± 0.01
5	0.93 ± 0.02	0.78 ± 0.02	0.89 ± 0.01
7	0.91 ± 0.01	0.80 ± 0.01	0.88 ± 0.01
8	0.87 ± 0.04	0.58 ± 0.05	0.79 ± 0.04
9	0.90 ± 0.01	0.78 ± 0.02	0.77 ± 0.01

To demonstrate the usability of these models in a real-time system, they were deployed on WHISPER [26], a real-time general purpose multi-channel audio platform with an ARM core, specifically a Raspberry Pi 2 Model b that features a 900 MHz quad-core ARM Cortex A7 and 1 GB of RAM.

Our proposed model for KWS is competitive with state-of-the-art models, e.g. [4, 5, 6]. It uses spike count frames that only incur a feature computation delay of $5\,\mathrm{ms}$ while more traditional methods that use log-filterbank features usually incur a feature computation delay of $30 - 40\,\mathrm{ms}$. Moreover, the algorithmic latency introduced by the number of frames needed to drive the model is between $100 - 200\,\mathrm{ms}$ for traditional approaches whereas the latency for this model is just $5\,\mathrm{ms}$ since only one frame per step is needed by our model. By running the model on the Raspberry Pi, we obtain a computation time of $974 \pm 0.14\,\mathrm{ms}$ to process 300 frames corresponding to $1500\,\mathrm{ms}$ of real time data. From this we can extrapolate that each frame takes about $3\,\mathrm{ms}$ to be processed and thus can allow for real time computation with a latency of $5\,\mathrm{ms}$. Since the SV model has a similar architecture, we can assume that the system latency will also be small.

The memory footprint of the entire system is 6 MB. It comprises the stored ITD priors for the localization model, and the weights for both models. In terms of hardware implementation, the system only needs an accumulator and a look-up table for the logarithmic compression in order to compute the spike features. It also bypasses the more costly computation of frequency-domain filtering for both feature computation and signal enhancement. Furthermore, the data-driven nature of the spikes is useful for hardware accelerators such as [27] that can skip computations if there is no input.

5. CONCLUSION

We present a novel low-resource extensive acoustic pipeline for a voice-activated smart assistant that uses as front end, an event-based low power audio sensor. The pipeline steps include localization and source separation of the target speaker along with keyword activation and speaker verification. Even though the performances on the single tasks are not at the level of state-of-the-art approaches, the pipeline provides multiple advantages including reduced system latency and computational cost. Further improvements in recognition rates can be obtained with the use of a larger dataset.

6. ACKNOWLEDGEMENT

This work was partially supported by the European Union's Horizon 2020 research and innovation program under grant agreement No 644732 and the Swiss National Science Foundation grant agreement No. 200021_172553.

References

[1] L. Lantian, T. Zhiyuan, W. Dong, A. Andrew, F. Yang, and Z. Shiyue, *Collaborative Learning for Language and Speaker Recognition*, Springer, Singapore, 2018.

[2] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification," *IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP*, 2016.

[3] Z. Shi, H. Lin, L. Liu, and R. Liu, "Double joint Bayesian modeling of DNN local i-vector for text dependent speaker verification with random digit strings," in *Interspeech*, 2018, pp. 2663–2667.

[4] S. Myer and V. S. Tomar, "Efficient keyword spotting using time delay neural networks," in *Interspeech*, 2018, pp. 1264–1268.

[5] R. Kumar, V. Yeruva, and S. Ganapathy, "On convolutional LSTM modeling for joint wake-word detection and text dependent speaker verification," in *Interspeech*, 2018, pp. 2663–2667.

[6] M. Chen, S. Zhang, M. Lei, Y. Liu, H. Yao, and J. Gao, "Compact feedforward sequential memory networks for small-footprint keyword spotting," in *Interspeech*, 2018, pp. 2663–2667.

[7] 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 *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015, pp. 4704–4708.

[8] V. Mitra, J. Van Hout, H. Franco, D. Vergyri, Y. Lei, M. Graciarena, Y. Tam, and J. Zheng, "Feature fusion for high-accuracy keyword spotting," in *IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP*, 2014.

[9] Y. Wang, P. Getreuer, T. Hughes, R. F. Lyon, and R. A. Saurous, "Trainable frontend for robust and far-field keyword spotting," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, 2017.

[10] Y. Huang, T. Hughes, T. Z. Shabestary, and T. Applebaum, "Supervised noise reduction for multichannel keyword spotting," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, 2018.

[11] S-C. Liu, A. van Schaik, B. A. Minch, and T. Delbruck, "Asynchronous binaural spatial audition sensor with $2\times64\times4$ channel output," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 8, no. 4, pp. 453–464, Aug 2014.

[12] V. Chan, S-C. Liu, and A. van Schaik, "AER EAR: A matched silicon cochlea pair with address event representation interface," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 54, no. 1, pp. 48–59, 2007.

[13] M. Yang, C.H. Chien, T. Delbruck, and S-C. Liu, "A 0.5V 55μW 64×2 channel binaural silicon cochlea for event-driven stereo-audio sensing," *IEEE Journal of Solid-State Circuits*, vol. 51, no. 11, pp. 2554–2569, 2016.

[14] J. Anumula, E. Ceolini, Z. He, A. Huber, and S-C. Liu, "An event-driven probabilistic model of sound source localization using cochlea spikes," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2018.

[15] E. Ceolini, J. Anumula, A. Huber, I. Kiselev, and S-C. Liu, "Speaker activity detection and minimum variance beamforming for source separation," in *Interspeech*, 2018.

[16] C. H. Li, T. Delbruck, and S. C. Liu, "Real-time speaker identification using the AEREAR2 event-based silicon cochlea," in *2012 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2012, pp. 1159–1162.

[17] M. Abdollahi and S. C. Liu, "Speaker-independent isolated digit recognition using an AER silicon cochlea," in *2011 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, Nov 2011, pp. 269–272.

[18] M. Yang, C. Yeh, Y. Zhou, J. P. Cerqueira, A. A. Lazar, and M. Seok, "A 1μW voice activity detector using analog feature extraction and digital deep neural network," in *2018 IEEE International Solid - State Circuits Conference - (ISSCC)*, Feb 2018, pp. 346–348.

[19] J. B. Allen and D. A. Berkley, "Image method for efficiently simulating small-room acoustics," *J. Acoust. Soc. Am*, vol. 65, no. 4, pp. 943–950, 1979.

[20] R. Leonard, "A database for speaker-independent digit recognition," in *IEEE International Conference on Acoustics, Speech, and Signal Processing ICASSP*, 1984.

[21] `https://github.com/SensorsAudioINI/caesar_icassp2019.git`.

[22] J. Anumula, D. Neil, T. Delbruck, and S-C. Liu, "Feature representations for neuromorphic audio spike streams," *Frontiers in Neuroscience*, vol. 12, pp. 23, 2018.

[23] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML Deep Learning Workshop*, 2015.

[24] J. Chung, G. Caglar, K.H. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *CoRR*, vol. abs/1412.3555, 2014.

[25] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist Temporal Classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd International Conference on Machine Learning*, 2006, pp. 369–376.

[26] I. Kiselev, E Ceolini, D. Wong, A. d. Cheveigne, and S-C. Liu, "WHISPER: Wirelessly synchronized distributed audio sensor platform," in *2017 IEEE 42nd Conference on Local Computer Networks Workshops (LCN Workshops)*, Oct 2017, pp. 35–43.

[27] C. Gao, D. Neil, E. Ceolini, S-C. Liu, and T. Delbruck, "DeltaRNN: A power-efficient recurrent neural network accelerator," in *Proceedings of the 2018 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, 2018, pp. 21–30.