Automatic Audio Feature Extraction for Keyword Spotting

Paola Vitolo, *Graduate Student Member, IEEE*, Rosalba Liguori, *Member, IEEE*, Luigi Di Benedetto, Alfredo Rubino, and Gian Domenico Licciardo, *Senior Member, IEEE*

***Abstract*—The accuracy and computational complexity of keyword spotting (KWS) systems are heavily influenced by the choice of audio features in speech signals. This letter introduces a novel approach for audio feature extraction in KWS by leveraging a convolutional autoencoder, which has not been explored in the existing literature. Strengths of the proposed approach are in the ability to automate the extraction of the audio features, keep its computational complexity low, and allow accuracy values of the overall KWS systems comparable with the state of the art. To evaluate the effectiveness of our proposal, we compared it with the widely-used Mel Frequency Cepstrum (MFC) method in terms of classification metrics in noisy conditions and the number of required operators, using the public Google speech command dataset. Results demonstrate that the proposed audio feature extractor achieves an average classification accuracy on 12 classes ranging from 81.84% to 90.36% when the signal-to-noise ratio spans from 0 to 40 dB, outperforming the MFC up to 5.2%. Furthermore, the required number of operations is one order of magnitude lower than that of the MFC, resulting in a reduction in computational complexity and processing time, which makes it well-suited for integration with KWS systems in resource-constrained edge devices.**

***Index Terms*—Autoencoder, edge computing, keyword spotting, neural network, speech feature extraction.**

I. Introduction

Keyword Spotting (KWS) has become a crucial topic in recent years as it allows voice recognition systems to be more responsive, saves energy, and improves privacy and data security in edge computing contexts [1], [2], [3]. The conventional KWS pipeline consists of three main building blocks: a preprocessing stage to adapt the microphone output to audio processing systems, a feature extraction block, and a Neural Network (NN)-based classifier. The extraction of feature information is crucial for audio classification as it directly affects identification accuracy as well as the overall computational complexity of the system. It represents a computationally complex block as it heavily relies on mel-scale-related techniques, such as Mel-Frequency Cepstral Coefficients (MFCCs), mel spectrogram, and Power Normalized Cepstral Coefficients (PNCCs), which require Fourier Transforms (FT) in various processing stages [4], [5]. Several researchers have recently delved into investigating speech feature lightening approaches, such as quantizing speech feature maps [6] or reducing MFCC feature matrices [7], [8]. However, these methods still rely on computationally complex FT processes.

The aim of this work is to demonstrate the advantages of using Auto-Encoders (AE) [9], [10], [11] in implementing an automated data-driven approach for audio feature extraction. The main advantages of the proposed approach are as follows:

- Automated feature extraction, eliminating the need for an expert to choose the best features and set the extractor parameters appropriately;
- Reduction in the number of required operators, leading to lower power consumption and processing time;
- Potential integration of the proposed extractor with the other NN-based blocks of KWS systems (e.g. with PDM-to-PCM converter [12] and NN-based classifier [13]), thereby creating a compact NN-based end-to-end KWS system.

Advantages of the proposed AE-based feature extractor stem from comparisons with the conventional MFCCs, which are the most used features in state-of-the-art KWS systems [14], [15]. Several aspects have been considered: the number of operations required, classification metrics when used for KWS classification, and noise behavior. The KWS classifier and the dataset used to evaluate the feature performance have been the model described in the MLCommons/tiny benchmarking system [13], [16] and the public Google Speech Command Dataset (GSCD) [17]. In absence of noise, the KWS classifier that uses the proposed extracted features achieves an accuracy of 90.36%, which is comparable to the counterpart. However, the proposed approach exhibits a 5.2% higher accuracy when the Signal-to-Noise Ratio (SNR) equals 0, making it more robust in noisy environments. Additionally, the number of operators of the proposed approach is one order of magnitude lower than that of the MFCC method, resulting in a reduction in computational complexity and required processing time.

II. The Proposed Audio Feature Extraction

Fig. 1 illustrates the proposed setup to realize the automatic audio feature extractor, where an encoder and decoder are combined to form an AE. The primary function of the encoder is to extract a compressed low-dimensional representation of the

Manuscript received 4 August 2023; revised 24 November 2023; accepted 17 December 2023. Date of publication 22 December 2023; date of current version 8 January 2024. This work was supported by the Italian Ministry of Education, University and Research. The associate editor coordinating the review of this manuscript and approving it for publication was Mr. Ville M. Hautamäki. *(Corresponding author: Gian Domenico Licciardo.)*

The authors are with the Department of Industrial Engineering, University of Salerno, 84084 Fisciano, Salerno, Italy (e-mail: pvitolo@unisa.it; rliguori@unisa.it; ldibenedetto@unisa.it; arubino@unisa.it; gdlicciardo@unisa.it).

Digital Object Identifier 10.1109/LSP.2023.3346280

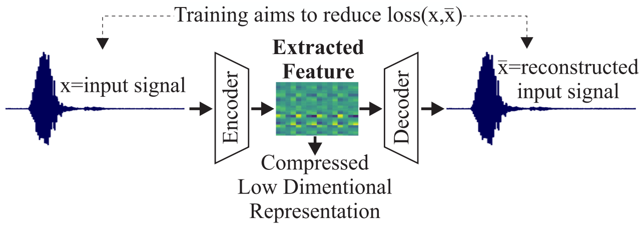

Training aims to reduce loss(x,x̄)
x=input signal
Encoder
Extracted Feature
Decoder
x̄=reconstructed input signal
Compressed Low Dimentional Representation

Fig. 1. Block diagram of the proposed automatic audio feature extractor. Network training minimizes input-output errors, allowing autonomous learning of robust features, reducing noise impact while preserving crucial information.

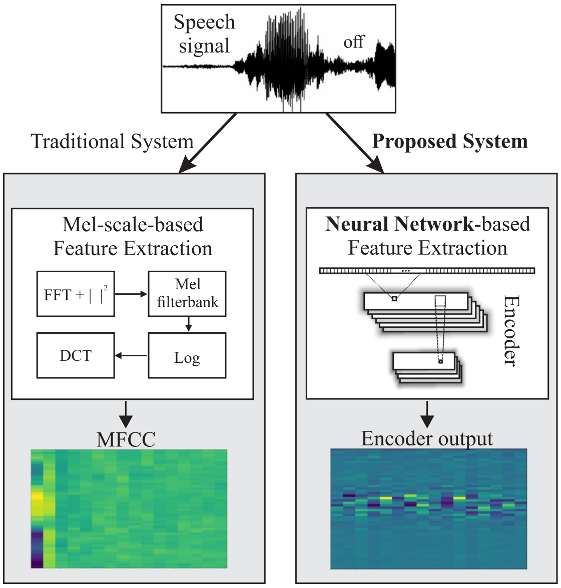


Fig. 2. Block Diagrams of the traditional mel-scale-based (left) and proposed NN-based (right) feature extraction systems.

TABLE I
SUMMARY OF THE PROPOSED MODEL

Layer	No. of channels	kernel/ pool size	No. of parameters	Output Shape
Conv2D_1	32	32×1	1056	16000×32
MaxPool2D_1	-	4×4	0	4000×8
Conv2D_2	16	8×8	1,040	4000×8×16
MaxPool2D_2	-	4×2	0	2000×32
Conv2D_3	2	16×16	514	2000×32×2
MaxPool2D_3	-	4×4	0	500×8×2
Conv2D_4	2	8×8	130	2000×4×2
MaxPool2D_4	-	5×4	0	50×16
Upscale2D	-	5×1	0	250×16
Conv2D_T	2	16×16	514	250×16×2
Conv2D_T	2	16×16	514	500×16×2

input data, while the decoder aims to reconstruct the input using the features extracted by the encoder. The AE is trained with the objective of minimizing the loss function between the input and output. In this way, while the AE learns to reconstruct the input, the encoder learns to automatically derive the most informative features from the input signal. Once the training is completed, the decoder can be discarded and the encoder alone is used as a feature extractor as shown in Fig. 2.

A. Neural Network Architecture

The architecture of the proposed AE is depicted in Fig 3. The AE consists of two main components: the encoder, which serves as a feature extractor and dimensional reducer, and the decoder, responsible for reconstructing the input from the extracted features. During training, the network tries to minimize input-output errors. This iterative process enables the network to autonomously learn robust features, effectively mitigating the impact of noise while retaining crucial information. The proposed model is based on a Convolutional Neural Network (CNN), chosen because of its efficiency and effectiveness in finding local spatial coherences, and its reduced number of parameters and required operations compared to other models [18], [19], [20]. While CNNs are widely known for their success in image processing [19], they have also shown promising results in speech-related applications, such as speech recognition, speaker identification, and emotion recognition [21]. While significant progress has been made in developing automated neural architecture search in recent years [22], identifying the optimal architecture remains a challenging task that continues to rely on specific case studies and expert knowledge. In this work, the proposed AE model has been designed through an iterative trial-and-error process, involving adjustments to the number of layers, the number of neurons per layer, and the hyperparameter values. The proposed encoder consists of four convolutional layers (CONV2D) that act as filters, followed by max-pooling layers (MaxPool) to reduce the input dimensionality. Conv2D_1, Conv2D_2, Conv2D_3, and Conv2D_4 have a kernel size of (32 × 1), (8 × 8),(16 × 16), and (8 × 8), respectively, while their channels are 32, 16, 2, and 2. The strides are equal to 1, the padding is set to “same”, and the activation functions have been the hyperbolic tangent (1). The pool size of the first three pooling layers is (4 × 4), while the size of the last one is (5 × 4).

$$y = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \quad (1)$$

The decoder is composed of 1 upsampling layer of (5 × 1) size and 2 transposed convolution layers with 2 channels, (1) as activation function, and a kernel size of (16 × 16). Table I reports the architecture of the proposed AE, detailing the layers, number of parameters, and output shapes. To assess the effectiveness of the proposed feature extractor (Fig. 2 right) in comparison to the traditional MFCC extraction (Fig. 2 left), a keyword classifier has been trained using features obtained from both extractors.

The chosen classifier follows the architecture proposed in the MLCommons/tiny benchmarking system [13], [16]. It consists of the following layers:

- One 2D convolutional layer with a kernel size of 10 × 4 and 64 channels.
- Four 2D depthwise separable convolution layers (DSConv2D).
- One 2D max pooling layer.
- One dense layer with 12 neurons.

Each DSConv2D layer is composed of a depthwise 2D convolutional layer with a kernel size of 3 × 3, followed by a 2D

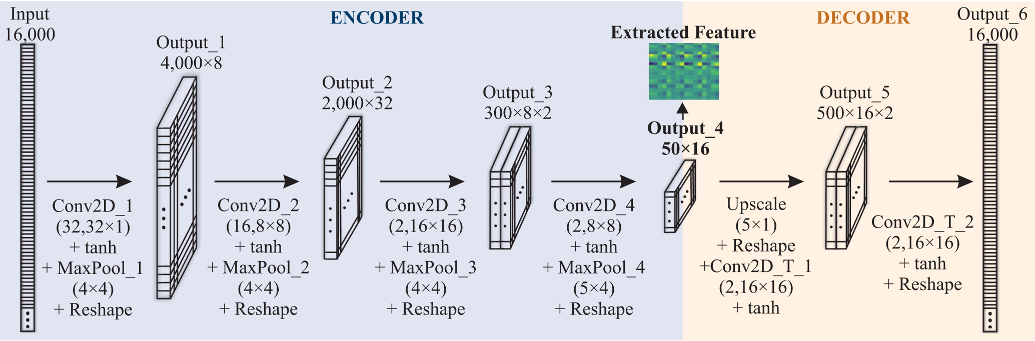


Fig. 3. Architecture of the proposed convolutional autoencoder. It consists of an encoder (left) and a decoder (right) part. The encoder acts as a feature extractor and dimensional reducer, while the decoder reconstructs the input from the extracted feature.

convolutional layer with a kernel size of 1 × 1 and 64 channels. The activation function used for the convolutional layers is the ReLU function (2), while the softmax function (3) is applied to the last layer to obtain probabilities for class membership.

$$y = \begin{cases} 0 & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases} \tag{2}$$

$$y(x)_j = \frac{e^{x_j}}{\sum_i e^{x_i}} \tag{3}$$

B. Dataset

The GSCD referenced in [17] has been used to train and evaluate the proposed feature extraction system. This dataset consists of 105829 utterances of 35 words. Each utterance is a one-second (or less) WAVE file, encoded with 16-bit single-channel PCM values and a sampling rate of 16 kHz [17]. Twelve specific classes have been selected from the dataset, following the approach used in the MLCommons/tiny benchmarking system [13], [16]. The chosen classes are: “Yes,” “No,” “Up,” “Down,” “Left,” “Right,” “On,” “Off,” “Stop,” “Go,” “Background,” and “Unknown.” The “Background” class comprises one-second clips randomly extracted from background noise audio files, while the “Unknown” class consists of randomly sampled words from the remaining classes. To ensure uniformity in the dataset, recordings shorter than 1 s have been zero-padded. Additionally, to create a balanced dataset, the same number of words has been selected for every class. As the class “Up” contains the fewest words, i.e., 3723 words [17], this number has been used for each class, resulting in a total dataset size of 44676 words. Additive white gaussian noise has been employed to evaluate the noisy robustness of the proposed approach, as it represents the lowest performance bound among all noise types [23]. The noise has been added to each utterance in the dataset, for a SNR ranging from 0 to 40 dB.

C. Training

The proposed autoencoder and the classifier proposed by [13], [16] described in Section II-A have been modeled and trained in Python language using TensorFlow (TF) framework [24]. For the autoencoder, two loss functions were employed: the Mean Absolute Error (MAE) and the Fast-Fourier-Transform Mean Absolute Error (FFT-MAE). The FFT-MAE is a custom loss function introduced in [25], [26]. As described by (4), it returns the mean absolute error between the FFT of the model outputs and the FFT of the corresponding labels. The model has been initially trained with MAE as the loss function. Subsequently, the model has been fine-tuned by using FFT-MAE.

$$FFT_{MAE} = \frac{1}{n} \sum_{i=0}^{n} ||FFT(Y_i)| - |FFT(\hat{Y}_i)|| \tag{4}$$

The dataset has been divided into training (80%), validation (10%), and test (10%) sets. The batch sizes and the number of epochs have been set to 64 and 100, respectively. Early stopping has been used to reduce overfitting, setting 0.6 patience and monitoring the validation loss. The performance of the proposed autoencoder has been evaluated in terms of MAE, Mean Square Error (MSE), and FFT-MAE. For the classifier, the dataset has been divided into training (80%) and test (20%) sets. A 4-fold cross-validation has been used to reduce overfitting and estimate the generalization performance of the model on different splits of the training dataset. The model has been trained with batch sizes of 256 for 200 epochs. Sparse categorical cross-entropy has been chosen as the loss function, Adam as the optimizer, and accuracy as the metric. The performance of the model has been evaluated in terms of accuracy, precision, recall, and f-score.

III. Results

The results of the autoencoder training show a MAE of 0.0338, a MSE of 0.00828, and FFT-MAE of 2.8 on the test set. Fig. 4 reports the classification metrics on the validation set of the classifier trained with the proposed extracted features for each fold. The accuracy averaged on the 4 folds of the classifier using the proposed features is 89.69%, with a standard deviation of 0.57, while the precision, recall, and f-score are 89.82% (0.59), 89.69% (0.51), 89.67% (0.58), respectively. These small standard deviation values ensure the robustness of the proposed model to various splitting data ensuring the reliability of the results. As can be seen from the confusion matrix in Fig. 5,

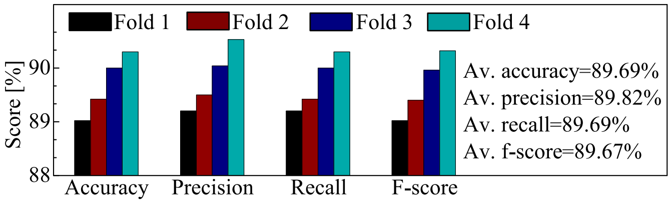
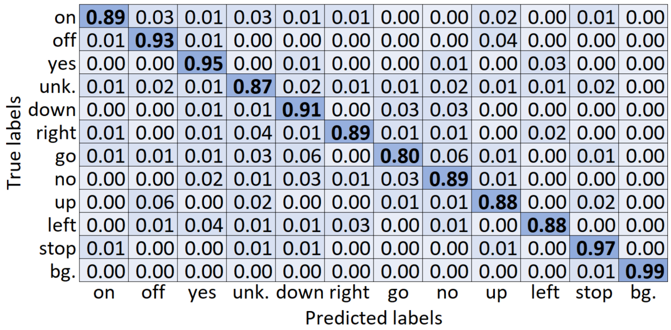


Fig. 4. K-fold cross-validation metrics of the proposed approach.

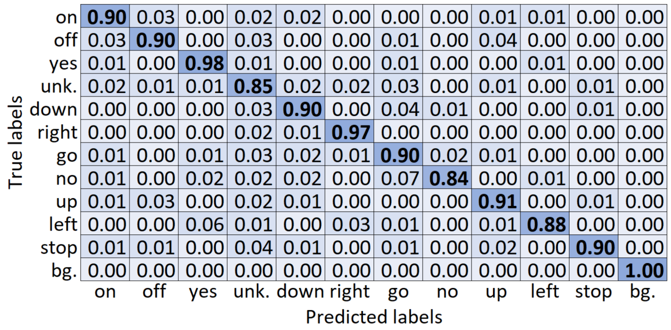

True labels
Predicted labels

	on	off	yes	unk.	down	right	go	no	up	left	stop	bg.
on	**0.89**	0.03	0.01	0.03	0.01	0.01	0.00	0.00	0.02	0.00	0.01	0.00
off	0.01	**0.93**	0.01	0.00	0.00	0.00	0.00	0.00	0.04	0.00	0.00	0.00
yes	0.00	0.00	**0.95**	0.00	0.01	0.00	0.00	0.01	0.00	0.03	0.00	0.00
unk.	0.01	0.02	0.01	**0.87**	0.02	0.01	0.01	0.02	0.01	0.01	0.02	0.00
down	0.00	0.00	0.01	0.01	**0.91**	0.00	0.03	0.03	0.00	0.00	0.00	0.00
right	0.01	0.00	0.01	0.04	0.01	**0.89**	0.01	0.01	0.00	0.02	0.00	0.00
go	0.01	0.01	0.01	0.03	0.06	0.00	**0.80**	0.06	0.01	0.00	0.01	0.00
no	0.00	0.00	0.02	0.01	0.03	0.01	0.03	**0.89**	0.01	0.00	0.00	0.00
up	0.00	0.06	0.00	0.02	0.00	0.00	0.01	0.01	**0.88**	0.00	0.02	0.00
left	0.00	0.01	0.04	0.01	0.01	0.03	0.00	0.01	0.00	**0.88**	0.00	0.00
stop	0.01	0.00	0.00	0.01	0.01	0.00	0.00	0.00	0.01	0.00	**0.97**	0.00
bg.	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	**0.99**

Fig. 5. Confusion matrix on the test set of the proposed extractor-based classifier.

TABLE II
ACCURACY ON THE TEST SET OF THE DIFFERENT METHODS

Feature	Mel Spectrogram	PNCC	MFCC	Proposed
Acc [%]	90.69	88.29	91.04	90.36


True labels
Predicted labels

	on	off	yes	unk.	down	right	go	no	up	left	stop	bg.
on	**0.90**	0.03	0.00	0.02	0.02	0.00	0.00	0.00	0.01	0.01	0.00	0.00
off	0.03	**0.90**	0.00	0.03	0.00	0.00	0.01	0.00	0.04	0.00	0.00	0.00
yes	0.01	0.00	**0.98**	0.01	0.00	0.00	0.01	0.00	0.00	0.01	0.00	0.00
unk.	0.02	0.01	0.01	**0.85**	0.02	0.02	0.03	0.00	0.01	0.00	0.01	0.00
down	0.00	0.00	0.00	0.03	**0.90**	0.00	0.04	0.01	0.00	0.00	0.01	0.00
right	0.00	0.00	0.00	0.02	0.01	**0.97**	0.00	0.00	0.00	0.00	0.00	0.00
go	0.01	0.00	0.01	0.03	0.02	0.01	**0.90**	0.02	0.01	0.00	0.00	0.00
no	0.01	0.00	0.02	0.02	0.02	0.00	0.07	**0.84**	0.00	0.01	0.00	0.00
up	0.01	0.03	0.00	0.02	0.01	0.00	0.00	0.00	**0.91**	0.00	0.01	0.00
left	0.00	0.00	0.06	0.01	0.00	0.03	0.01	0.00	0.01	**0.88**	0.00	0.00
stop	0.01	0.01	0.00	0.04	0.01	0.00	0.01	0.00	0.02	0.00	**0.90**	0.00
bg.	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**1.00**

Fig. 6. Confusion matrix on the test set of the classifier using the MFCCs.

the classifier using the proposed features achieves an average accuracy, precision, recall, and f-score on the test set of 90.36%, 90.51%, 90.36%, and 90.35%, respectively. The proposed approach has been compared to the following state-of-the-art features: mel spectrogram, MFCC, and PNCC [5], [27], [28]. As can be seen in Table II, the classifier using the proposed features achieves an accuracy approximately 2.07% greater than that based on PNCC while 0.68% and 0.33% lower than that obtained with MFCC and mel spectrogram, respectively. As can be seen from the confusion matrix on the test set in Fig. 6, the MFCC classifier obtains the best performance with an accuracy average over the 12 classes of 91.04% and a standard deviation of 0.045. However, as shown in Fig 7(a), the average accuracy of the MFCC classifier drops faster than the accuracy of the proposed solution as the SNR decreases. Specifically, with an SNR of 0 dB, the classifier using MFCC achieves an accuracy of 76.64%, which is 5.2% lower than the accuracy obtained by our classifier. Table III provides a comparison between the number of operations per frame required by the proposed feature extractor and the best one of Table II, MFCC calculated as shown in [29]. The proposed solution requires 2800 additions and 2688 multiplications per kernel size, which are about one order of magnitude lower than the counterpart (14192 additions and 12529 multiplications), and does not require the calculation of the Log function. The overall advantage of the proposed approach is shown in Fig 7(b), where the accuracy weighted for the number of required operations per frame of the proposed approach outperforms the MFCC for each SNR value. Therefore, the proposed solution can replace the resource-hungry FT-based feature extractor for more lightweight audio processing. These results pave the way to the design of a complete pipeline of neural networks, forming an end-to-end KWS system by merging the proposed feature extractor with the NN-based PDM-to-PCM converter in [12] and a CNN-based classifier.

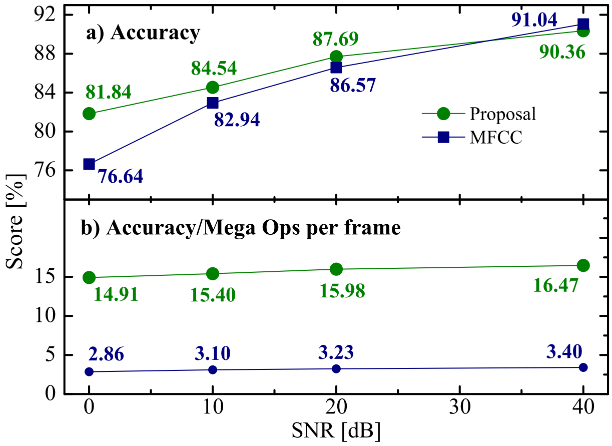


Fig. 7. Accuracy trends of the MFCC-based classifier and the proposed extractor-based classifier as SNR varies.

TABLE III
OPERATIONS PER FRAME REQUIRED BY THE PROPOSED FEATURE EXTRACTOR AND MFCC

		Add/Sub	Mult	Log
MFCC	FFT	3458	1729	0
	Mel Filterbank	9950	10000	0
	Mel Coefficient	784	800	50
	Tot	**14192**	**12529**	**50**
Proposed Feature Extractor	Conv+MaxPool1	1072	1024	0
	Conv+MaxPool2	1048	1024	0
	Conv+MaxPool3	530	512	0
	Conv+MaxPool4	150	128	0
	Tot	**2800**	**2688**	**0**

IV. CONCLUSION

This letter introduces a novel data-driven method using a compact AE for automated audio feature extraction. The evaluation in a KWS system, against MFCCs and using GSCD, has shown its advantages in accuracy, noise behaviour, and computational efficiency, ideal for resource-constrained devices. Future works will aim to integrate the proposed extractor into an end-to-end NN-based KWS system and design edge computing-specific hardware.

References

[1] Y. Zhang, N. Suda, L. Lai, and V. Chandra, "Hello edge: Keyword spotting on microcontrollers," 2017, *arXiv:1711.07128.*

[2] S. Sridhar and M. E. Tolentino, "Evaluating voice interaction pipelines at the edge," in *Proc. IEEE Int. Conf. Edge Comput.*, 2017, pp. 248–251.

[3] A. D. Vita, D. Pau, C. Parrella, L. D. Benedetto, A. Rubino, and G. D. Licciardo, "Low-power hwaccelerator for AI edge-computing in human activity recognition systems," in *Proc. IEEE 2nd Int. Conf. Artif. Intell. Circuits Syst.*, 2020, pp. 291–295.

[4] I. López-Espejo, Z.-H. Tan, J. H. L. Hansen, and J. Jensen, "Deep spoken keyword spotting: An overview," *IEEE Access*, vol. 10, pp. 4169–4199, 2022.

[5] B. H. Iswanto, H. Hafizhahullah, H. F. Pardede, and A. Zahra, "The power-normalized cepstral coefficient (PNCC) for convolutional neural networks-based robust speech command recognition," *J. Phys.: Conf. Ser.*, vol. 2596, no. 1, Sep. 2023, Art. no. 012021, doi: 10.1088/1742-6596/2596/1/012021.

[6] M. Luo, D. Wang, X. Wang, S. Qiao, and Y. Zhou, "Error-diffusion based speech feature quantization for small-footprint keyword spotting," *IEEE Signal Process. Lett.*, vol. 29, pp. 1357–1361, 2022.

[7] A. Riviello and J.-P. David, "Binary speech features for keyword spotting tasks," in *Proc. Interspeech*, 2019, pp. 3460–3464.

[8] I. López-Espejo, Z.-H. Tan, and J. Jensen, "An experimental study on light speech features for small-footprint keyword spotting," in *Proc. Iber-SPEECH*, 2022, pp. 131–135.

[9] P. Vitolo, A. D. Vita, L. D. Benedetto, D. Pau, and G. D. Licciardo, "Low-power detection and classification for in-sensor predictive maintenance based on vibration monitoring," *IEEE Sensors J.*, vol. 22, no. 7, pp. 6942–6951, Apr. 2022.

[10] A. Shrestha and A. Mahmood, "Review of deep learning algorithms and architectures," *IEEE Access*, vol. 7, pp. 53040–53065, 2019.

[11] P. Vitolo, G. D. Licciardo, L. d. Benedetto, R. Liguori, A. Rubino, and D. Pau, "Low-power anomaly detection and classification system based on a partially binarized autoencoder for in-sensor computing," in *Proc. 28th IEEE Int. Conf. Electron., Circuits, Syst.*, 2021, pp. 1–5.

[12] P. Vitolo, R. Liguori, L. D. Benedetto, A. Rubino, D. Pau, and G. D. Licciardo, "A new NN-based approach to in-sensor PDM-to-PCM conversion for ultra TinyML KWS," *IEEE Trans. Circuits Syst. II: Exp. Briefs*, vol. 70, no. 4, pp. 1595–1599, Apr. 2023.

[13] MLCommons, "Mlperf tiny benchmark suite," 2022. [Online]. Available: https://github.com/mlcommons/tiny/tree/master/benchmark

[14] Z. K. Abdul and A. K. Al-Talabani, "Mel frequency cepstral coefficient and its applications: A review," *IEEE Access*, vol. 10, pp. 122136–122158, 2022.

[15] P. Durairaj and S. Sriuppili, "Speech processing: MFCC based feature extraction techniques- an investigation," . *J. Phys.: Conf. Ser.*, vol. 1717, 2021, Art. no. 012009.

[16] V. J. Reddi, C. Cheng, D. Kanter, P. Mattson, G. Schmuelling, and C.-J. Wu, "The vision behind MLPerf: Understanding AI inference performance," *IEEE Micro*, vol. 41, no. 3, pp. 10–18, May/Jun. 2021.

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

[18] A. D. Vita, D. Pau, L. D. Benedetto, A. Rubino, F. Pétrot, and G. D. Licciardo, "Low power tiny binary neural network with improved accuracy in human recognition systems," in *Proc. 23 rd Euromicro Conf. Digit. System Des.*, 2020, pp. 309–315.

[19] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: Analysis, applications, and prospects," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 12, pp. 6999–7019, Dec. 2022.

[20] A. De Vita, A. Russo, D. Pau, L. D. Benedetto, A. Rubino, and G. D. Licciardo, "A partially binarized hybrid neural network system for low-power and resource constrained human activity recognition," *IEEE Trans. Circuits Syst. I: Regular Papers*, vol. 67, no. 11, pp. 3893–3904, Nov. 2020.

[21] A. Mehrish, N. Majumder, R. Bharadwaj, R. Mihalcea, and S. Poria, "A review of deep learning techniques for speech processing," *Inf. Fusion*, vol. 99, 2023, Art. no. 101869. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1566253523001859

[22] K. T. Chitty-Venkata, M. Emani, V. Vishwanath, and A. K. Somani, "Neural architecture search benchmarks: Insights and survey," *IEEE Access*, vol. 11, pp. 25217–25236, 2023.

[23] Y. Wang, Y. S. Chong, W. L. Goh, and A. T. Do, "Noise-aware and lightweight LSTM for keyword spotting applications," in *Proc. 19th Int. SoC Des. Conf.*, 2022, pp. 135–136.

[24] M. Abadi et al., "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," 2016, *arXiv:1603.04467*.

[25] P. Vitolo, R. Liguori, L. D. Benedetto, A. Rubino, D. Pau, and G. D. Licciardo, "A 0.8 mW tinyML-based PDM-to-PCM conversion for in-sensor KWS applications," in *Proc. Annu. Meeting Ital. Electron. Soc.*, 2023, pp. 146–151.

[26] P. Vitolo et al., "Quantized ID-CNN for a low-power PDM-to-PCM conversion in tinyML KWS applications," in *Proc. IEEE 4th Int. Conf. Artif. Intell. Circuits Syst.*, 2022, pp. 154–157.

[27] X. Liu, M. Sahidullah, and T. Kinnunen, "Optimized power normalized cepstral coefficients towards robust deep speaker verification," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2021, pp. 185–190.

[28] M. Turab, T. Kumar, M. Bendechache, and T. Saber, "Investigating multi-feature selection and ensembling for audio classification," *Int. J. Artif. Intell. Appl.*, vol. 13, no. 3, May 2022, doi: 10.5121/ijaia.2022.13306.

[29] J.-C. Wang, J.-F. Wang, and Y.-S. Weng, "Chip design of MFCC extraction for speech recognition," *Integration*, vol. 32, no. 1, pp. 111–131, 2002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167926002000457

Open Access provided by 'Università degli Studi di Salerno' within the CRUI CARE Agreement