Ultra-Low Power Keyword Spotting at the Edge

Mehmet Gorkem Ulkar
Analog Devices Inc.
Istanbul, Turkey
gorkem.ulkar@analog.com

Osman Erman Okman
Analog Devices Inc.
Istanbul, Turkey
erman.okman@analog.com

Abstract—**Keyword spotting (KWS) has become an indispensable part of many intelligent devices surrounding us, as audio is one of the most efficient ways of interacting with these devices. The accuracy and performance of KWS solutions have been the main focus of the researchers, and thanks to deep learning, substantial progress has been made in this domain. However, as the use of KWS spreads into IoT devices, energy efficiency becomes a very critical requirement besides the performance. We believe KWS solutions that would seek power optimization both in the hardware and the neural network (NN) model architecture are advantageous over many solutions in the literature where mostly the architecture side of the problem is considered. In this work, we designed an optimized KWS CNN model by considering end-to-end energy efficiency for the deployment at MAX78000, an ultra-low-power CNN accelerator. With the combined hardware and model optimization approach, we achieve 96.3% accuracy for 12 classes while only consuming 251 uJ per inference. We compare our results with other small-footprint neural network-based KWS solutions in the literature. Additionally, we share the energy consumption of our model in power-optimized ARM Cortex-M4F to depict the effectiveness of the chosen hardware for the sake of clarity.**

Index Terms—**speech commands, keyword spotting, wake word detection, convolutional neural networks, once-for-all, neural architecture, mfcc**

I. INTRODUCTION

Speech-based interfaces have been commonly used for smart devices like intelligent loudspeakers or mobile assistants, especially with the recent improvements in deep neural network (DNN) technologies. Large vocabulary speech recognition systems run big networks that require a lot of resources and high processing power; therefore, they are generally built as cloud-based solutions. On-device keyword spotting (KWS) is used in such systems to activate or support a limited number of interactions (i.e., commands) to reduce standby energy and latency, preserve user privacy, and avoid data leakage during data transmission.

Due to the low-latency requirement of KWS systems, many researchers focused on solutions with a small footprint in terms of computation and parameters. The authors in [1] proposed using a CNN with limited size instead of a DNN for the KWS application. In [2], the authors proposed the use of residual layers and dilations in CNN to increase the accuracy while satisfying the small footprint requirement. Graph convolutional networks are proposed in [3] to capture

non-local relations in KWS. Zhang *et al.* proposed a depthwise-separable CNN for MCU deployment in [4]. All of these mentioned methods use Mel-Frequency Cepstrum Coefficients (MFCC) to preprocess the raw audio first and then run their models on these features. While they are competing to get more compact and accurate models, the computation and power consumed in the MFCC remain stable. The authors in [5] proposes using Tsetlin Machines instead of NN to run on MFCC features, and they show that MFCC consumes most of the total energy of their end-to-end KWS system. This illustrates the significance of eliminating MFCC or other similar preprocessing from the KWS pipeline. Authors in [6] proposes using Sinc-convolutions in the feature extraction layer of their CNN network instead of MFCC. However, Sinc-convolutions are not supported by most of the known NN accelerators, if not none. Therefore this block needs to be implemented in the MCU if edge deployment is targeted.

Contrary to most literature where the number of operations and parameter number are the main concerns, energy is the most vital consideration for any battery-powered edge solution. With the introduction of NN accelerators, the metrics like multiply-accumulate operations (MAC) do not substitute for energy usage as the energy usage of those operations differs vastly depending on whether those operations can be done in an ultra-efficient NN accelerator or not. The main focus of this paper is to propose a highly accurate and ultra-energy-efficient KWS solution. For the lowest possible end-to-end energy usage, both the hardware and model architecture need to be optimized as shown in Fig. 1. The utilization of specialized low-power hardware for neural network inference is the general solution if custom hardware is not to be designed for the given application. However, most of the time, neural networks are not the single components of the keyword spotting systems. Preprocessing the audio signal or postprocessing the model outputs could be required besides the neural network inference operation. Removing those parts that cannot be run by an efficient accelerator and delegating their tasks to neural network blocks would be a viable method for increasing overall energy efficiency. Another room for improvement in the architecture side is to obtain the best architecture in terms of performance and energy for the given limitations of the deployment hardware.

In this study, we propose a CNN-based neural network architecture for KWS problems, ==which reduces the required energy consumption by removing the commonly used Mel-==

Spectrum transformation and utilizing an optimized neural network accelerator, MAX78000. We also employ a neural architecture search (NAS) approach and modify due to the limitations of the specialized hardware to optimize the accuracy of the final model. To summarize, our contributions can be listed as follows:

- In the hardware side, an ultra-low-power CNN accelerator, MAX78000, is utilized to implement the solution to present the overall energy reduction.
- In the NN architecture front, a network architecture is proposed for KWS applications that run on time-domain signals to reduce the required computation resources while preserving the accuracy.
- Once-For-All (OFA) based NAS approach is implemented and used to maximize the performance of the proposed network architecture while considering the MAX78000 hardware limitations.
- Proposed solution is compared with other keyword spotting methods in the literature regarding performance and energy consumption.
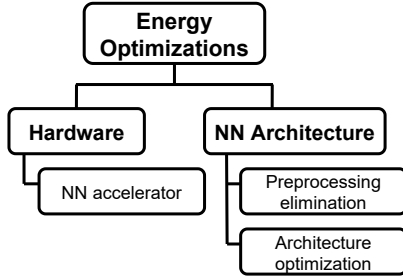

Fig. 1. Energy optimization methods

II. NEURAL NETWORK ARCHITECTURE

A. CNN Model with Raw Audio Input

In many of the keyword spotting applications, MFCC is commonly utilized features [1], [2], [4] which represents the speech signal with a set of known and relevant components. MFCC are achieved by decomposition of the audio signal using a filter bank. It provides the Discrete Cosine Transform (DCT) of a real logarithm of the short-term energy on the Mel frequency scale. More specifically, the computation pipeline of MFCC includes windowing the speech signal into frames, performing Fast Fourier Transform (FFT) to find the power spectrum of each frame, filter bank processing using the Mel scale, and finally DCT on the log scale of the power spectrum. Therefore, the computational load of this preprocessing step is generally much higher than the inference of the trained networks.

If the models that require MFCC processing are deployed to NN accelerators, MFCC has to be implemented in a general-purpose MCU outside of the accelerator, as shown in Fig. 2 B. However, this reduces the overall efficiency as an important portion of computation does not take place in the ultra-low-power NN accelerator. There are also some efforts in
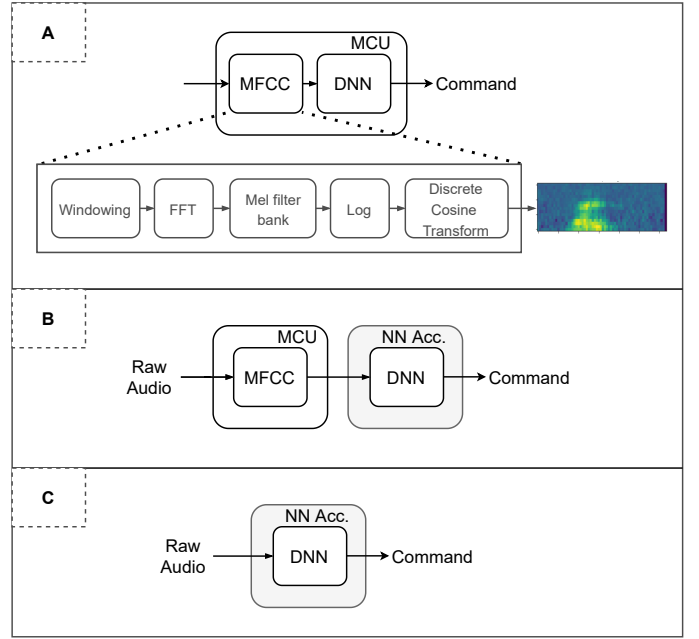

Fig. 2. KWS approaches

the literature to reduce the power consumption of MFCC processing, but it still requires more than two times more power than the neural network inference [7].

In this study, we propose to minimize the preprocessing step by elimination of the transformation to generate MFCC to reduce energy consumption effectively, as shown in Fig. 2 C. The raw audio signals are fed to the neural network after reshaping the signal, where the short-term and long-term variations can be observed. We used 1.024s length utterances sampled at 16kHz, and each of them are reshaped to 128x128x1 by folding the raw audio signal to the signal channels as shown in Fig. 3. Note that this strategy has no processing cost for our NN accelerator as it can be achieved by changing the order of the memory locations where the streaming input data is copied.

The advantage of this data structure is using 1D convolutional operators it is possible to extract features that represent both short time and long time relations. Therefore, in this study, we designed a base network seen in Fig. 4 which is composed of six 1-D convolutional layer units separated by pooling layers and a linear layer at the end. The units are composed of two or three convolutional layers is defined as 128 channel 5x5 kernels, and will then be modified by the NAS as described in the following section.

B. Neural Architecture Search

Neural architecture search (NAS) is an increasingly popular research topic in the literature [8]. By automating the exploration process, neural architecture search (NAS) methods aim to find the best neural architecture for a given set of requirements. The models that are found by contemporary NAS methods are difficult to obtain using manual design in
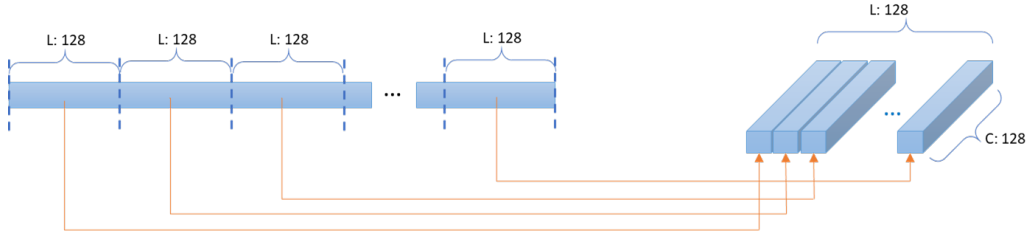
Fig. 3. Proposed data folding strategy. 16384x1 signal is reshaped to 128x128x1.

III. Experiments & Results

A. Dataset

terms of complexity and achieved accuracy. In this paper, we employed a weight-sharing-based NAS technique, Once-For-All (OFA) [9]. The authors in [9] propose training a super-network and deploying parts of it to diverse hardware without any need for retraining. This super-network is called "Once-for-All" network, and it requires a training process where all sub-networks are trained sufficiently to be deployed directly. Since training all sub-networks is computationally prohibitive, the authors came up with an idea of progressive shrinking. In this training method, the training starts with training the super-network with maximum kernel size, depth and width first and smaller sub-networks that share parameters with super-network become available for sampling progressively. By doing so, the training of interfering sub-networks at the same time is avoided. If the search space consists, different kernel, depth, and width values, they are added to sampling space sequentially. Once the whole training step is completed, the evolutionary search is performed to find the best sub-network for a given set of requirements. The constraints of targeted NN accelerator like memory are considered in this search step. In the Fig. 4, the designed super-network for the KWS is shown. This super-network contains convolutional layers with maximum width and kernel sizes, and the depth of the network is set to maximum. Specifically, the designed super-network contains 14 identical Conv1D layers with 128 channels and 5 as kernel lengths. Upon OFA training and evolutionary search, the sub-network selected automatically can be shown in the same Fig. 4 with parameter overrides in green font. In case any layer of the automatically selected network has the same parameter value with the super network, it is not indicated with green, so the red default values are used. It is noteworthy that the sub-network is much smaller than the super-network where its parameters are shared with. Another observation is that in this case, OFA did not prefer to eliminate any layer from the super network, although it could do so. Rather than decreasing the depth of the model, it chooses to slim the layers or shorten the kernels.

According to the OFA scheme, the model is ready to deploy after training and search steps. However, the model needs to be quantized if it is to be deployed to an NN accelerator or any MCU. For this purpose, we employed quantization-aware training (QAT) rather than post-quantization as QAT yields networks with higher performances.

All the codes that have been developed for this study are open-sourced and can be found at [10].

The proposed approach is evaluated using the Google Speech Commands Dataset v2 [11], which contains 65000 1-second long utterances of 35 short words by thousands of different people, as well as some background noises. Based on the recent researches for KWS applications [2], [4], [12], [13], the dataset is relabeled to form 12 classes: "Yes", "No", "Up", "Down", "Left", "Right", "On", "Off", "Stop", "Go" along with the "Silence" which contains no speech utterances and the "Unknown" which contains the data from the remaining 25 keywords in the original dataset. The dataset is split into training, validation, and test set in the ratio of 80:10:10 while the audio clips from the same person are kept in the same set.

An augmentation procedure is also followed to generate the training data to increase the robustness and the performance of the neural network model. This procedure is composed of signal stretching, utterance shifting, and noise addition, respectively. As a first step, the utterance is stretched with a random factor selected from $\mathcal{U}(0.8, 1.3)$. A random time shift, $t\ ms$, is then applied to the audio signal, where $t \sim \$(\mathcal{U})(-0.1, 0.1)$. Lastly, a zero-mean white Gaussian noise is added to the audio signal with variance σ^2, which is a uniformly distributed random variable between 0 and 1.

During training, every sample in the dataset is used, so there are more samples from the "Unknown" class than the other class samples. In order to prevent the model from getting biased, the class weight of the "Unknown" samples is reduced proportionally to the amount of the other class samples.

B. Model Training

In this study, model training includes two steps. As the first step, the supernet given in Fig. 4 is trained using Stochastic Gradient Descent (SGD) optimizer with a learning rate of 0.001 for 15000 epochs. Knowledge distillation is also used during the elastic training stages as described in the original OFA paper [9]. It fuses two loss terms using both the soft labels of the supernet just before initialization of the elastic training stages, and the real labels [14]. In Fig. 5, the progress of the training is given in terms of the objective loss with respect to both training epochs and time spent. The elastic kernel, elastic depth, and elastic width stages are also given in this graph. An evolutionary search algorithm is run using this model to find the best-performed model given in Fig. 5. Note
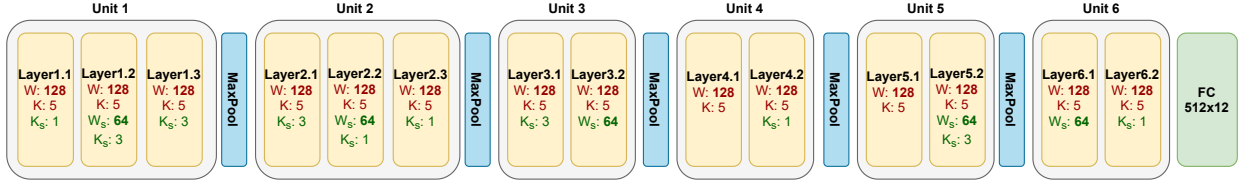
Fig. 4. KWS super-network for OFA

that the details of the search algorithm and selected parameter values are given in our project repository [10].

The selected model parameters must be quantized to at most 8-bits integers in order to be deployed to our edge platform described in the following section. However, post-quantization of the trained networks causes high-performance degradation, and it is much higher when the model is deeper or includes batch normalization, according to our experiments. Therefore, we use the second step of training, Quantization Aware Training (QAT) of the selected model, to fine-tune the model parameters while minimizing the induced errors due to the quantization. The QAT approach implements the fake (simulated) quantization approach in [15]. In this training step, an Adam optimizer is used with an initial learning rate of 0.001, gradually halved 5 times starting from the 100th epoch. The model is trained for 200 epochs, and the QAT procedure is initiated after epoch 150. In this study, all model parameters are quantized to 8-bits, and existing batch normalization layers are folded into the weights with the initialization of QAT.
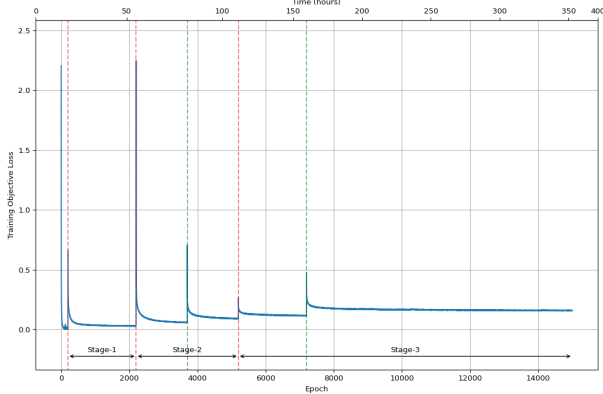

Fig. 5. Objective Loss vs. Number of Epochs for training OFA supernet

C. Neural Network Accelerator

As a deployment platform of the proposed CNN model, MAX78000 is chosen. MAX78000 is an Ultra-Low-Power Arm Cortex-M4 processor with FPU-based MCU with CNN accelerator. By spending only microjoules of energy to execute inferences, MAX78000 targets battery-powered neural network-based applications at the edge [16]. The CNN engine supports both Conv1D and Conv2D as convolution operations, 2D transposed convolution layer, linear layer, pooling layers, and element-wise operations as addition, subtraction, binary OR, binary XOR.

Model	MFCC	Precision	#Param	Model Size	Acc
[1]	yes	fp-32	244.2K	976.8 KB	90.2%
[2]	yes	fp-32	238K	952 KB	95.8%
[3]	yes	fp-32	61K	244 KB	96.4%
[4]	yes	int-8	497.6K	497.6 KB	95.4%
[6]	no	fp-32	122K	488 KB	96.6%
ours	no	**int-8**	419.8K	**419.8 KB**	**96.3%**

D. Accuracy Results

Table I summarizes the results of our findings of the proposed KWS approach with the selected small footprint models in the literature. We use class accuracy as the evaluation metric to be consistent with the literature. However, these studies consider the footprint of the approach by only taking the model complexity into account without considering the feature extraction step, i.e., MFCC calculation. Therefore, we present the table with a field that shows whether the approach uses MFCC or not. The number of parameters in the model, as well as the precision of those parameters and activations, are given in the table.

The proposed model in this study provides an accuracy of over 96%, which is above or comparable with the state-of-the-art small footprint KWS models despite not utilizing pre-calculated MFCCs. Having similar accuracies with the state-of-the-art proves that specialized hardware for Neural Net inference can easily be used to run the pre-trained models for KWS applications on the edge as the model runs on the raw audio signal.

E. Energy Measurements

In order to analyze the efficiency provided by MAX78000 and our approach, two sets of measurements are presented in this study. The first measurement is made to calculate the latency and the energy consumption of the MAX78000 for the proposed model. The same model is also implemented for an ultra-low-power ARM Cortex M4F microprocessor, and the measurements are given in II. It should be noted that the same model consumes more than 40 times more energy when deployed in the ARM Cortex M4F processor compared to MAX78000 CNN accelerator. One major element in this low consumption figure is that our model does not need any preprocessing outside the accelerator; therefore, it can better leverage the accelerator's energy efficiency.

TABLE II
COMPARISON OF LATENCY AND ENERGY MEASUREMENTS OF THE
PROPOSED MODEL FOR DIFFERENT HARDWARE

Device	Latency (ms/inf)	Energy (uJ/inf)
MAX78000	3.5	251
ARM Cortex M4F	905	11200

Secondly, we measure the energy and latency for MFCC calculation to show the efficiency of the proposed approach for the state-of-the-art approaches as they use MFCC transformation as an initial feature extraction procedure. 40 MFCC features extracted from a speech frame of length 40ms with a stride of 20ms, ending with a 49x40 data for a 1s long audio. Note that this is the same configuration used in [4] and similar to the others used in the literature. The described MFCC feature extraction is measured to consume 826 uJ, more than three times what our end-to-end KWS model consumes in the MAX78000 accelerator. This means any conventional model in the literature that is built upon MFCC features consumes much more energy than our solution, even if it is deployed in the convolution efficient hardware.

IV. CONCLUSION

In this paper, we present a highly accurate and energy-efficient KWS solution that runs on battery-powered devices. For this purpose, we propose a novel CNN model that works on raw audio by eliminating commonly used MFCC preprocessing. By doing so, the whole of the proposed network becomes deployable to the NN accelerators. In order to find the best architecture, we adopted a weight-sharing-based NAS approach, OFA, and we show that our accuracy is comparable with the state-of-the-art small footprint KWS solutions. Lastly, we deploy the model to a NN accelerator, MAX78000, and present its effectiveness in terms of energy and latency.

V. ACKNOWLEDGEMENTS

The authors would like to thank Robert Muchsel, Brian Rush and other members from the AI Development Group at Analog Devices that contributed to this work.

REFERENCES

[1] T. N. Sainath and C. Parada, "Convolutional neural networks for small-footprint keyword spotting," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[2] R. Tang and J. Lin, "Deep residual learning for small-footprint keyword spotting," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 5484–5488.

[3] X. Chen, S. Yin, D. Song, P. Ouyang, L. Liu, and S. Wei, "Small-footprint keyword spotting with graph convolutional network," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2019, pp. 539–546.

[4] Y. Zhang, N. Suda, L. Lai, and V. Chandra, "Hello edge: Keyword spotting on microcontrollers," 2018.

[5] J. Lei, T. Rahman, R. Shafik, A. Wheeldon, A. Yakovlev, O.-C. Granmo, F. Kawsar, and A. Mathur, "Low-power audio keyword spotting using tsetlin machines," *Journal of Low Power Electronics and Applications*, vol. 11, no. 2, 2021. [Online]. Available: https://www.mdpi.com/2079-9268/11/2/18

[6] S. Mittermaier, L. Kürzinger, B. Waschneck, and G. Rigoll, "Small-footprint keyword spotting on raw audio data with sinc-convolutions," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7454–7458.

[7] W. Shan, M. Yang, T. Wang, Y. Lu, H. Cai, L. Zhu, J. Xu, C. Wu, L. Shi, and J. Yang, "A 510-nw wake-up keyword-spotting chip using serial-fft-based mfcc and binarized depthwise separable cnn in 28-nm cmos," *IEEE Journal of Solid-State Circuits*, vol. 56, no. 1, pp. 151–164, 2021.

[8] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, X. Chen, and X. Wang, "A comprehensive survey of neural architecture search: Challenges and solutions," 2021.

[9] H. Cai, C. Gan, T. Wang, Z. Zhang, and S. Han, "Once for all: Train one network and specialize it for efficient deployment," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://arxiv.org/pdf/1908.09791.pdf

[10] "Max7800x model training," https://github.com/MaximIntegratedAI/ai8x-training, 2021.

[11] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," 2018.

[12] R. Tang and J. Lin, "Honk: A pytorch reimplementation of convolutional neural networks for keyword spotting," 2017.

[13] T. Mo, Y. Yu, M. Salameh, D. Niu, and S. Jui, "Neural Architecture Search for Keyword Spotting," in *Proc. Interspeech 2020*, 2020, pp. 1982–1986. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2020-3132

[14] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015.

[15] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[16] *MAX78000 Datasheet - Artificial Intelligence Microcontroller with UltraLow-Power Convolutional Neural Network Accelerator*, Maxim Integrated, 5 2021, rev. 1.