IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 10, 2022, accepted February 20, 2022, date of publication February 28, 2022, date of current version March 21, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3155124

Few-Shot User-Definable Radar-Based Hand Gesture Recognition at the Edge

GIANFRANCO MAURO[1,2], MATEUSZ CHMURSKI[1,3], LORENZO SERVADEI[1,4], (Member, IEEE), MANUEL PEGALAJAR-CUELLAR[2], AND DIEGO P. MORALES-SANTOS[2]

[1]Infineon Technologies AG, 85579 Neubiberg, Germany
[2]Department of Electronic and Computer Technology, University of Granada, 18071 Granada, Spain
[3]Department of Microelectronics and Computer Science, Lodz University of Technology, 90924 Łódź, Poland
[4]Department of Electrical and Computer Engineering, Technical University of Munich, 80333 Munich, Germany

Corresponding author: Gianfranco Mauro (gianfranco.mauro@infineon.com)

This work was supported in part by ITEA3 Unleash Potentials in Simulation (UPSIM) by the German Federal Ministry of Education and Research (BMBF) under Project 19006, in part by the Austrian Research Promotion Agency (FFG), in part by the Rijksdienst voor Ondernemend Nederland (Rvo), and in part by the Innovation Fund Denmark (IFD).

ABSTRACT Technological advances and scalability are leading Human-Computer Interaction (HCI) to evolve towards intuitive forms, such as through gesture recognition. Among the various interaction strategies, radar-based recognition is emerging as a touchless, privacy-secure, and versatile solution in different environmental conditions. Classical radar-based gesture HCI solutions involve deep learning but require training on large and varied datasets to achieve robust prediction. Innovative self-learning algorithms can help tackling this problem by recognizing patterns and adapt from similar contexts. Yet, such approaches are often computationally expensive and hardly integrable into hardware-constrained solutions. In this paper, we present a gesture recognition algorithm which is easily adaptable to new users and contexts. We exploit an optimization-based meta-learning approach to enable gesture recognition in learning sequences. This method targets at learning the best possible initialization of the model parameters, simplifying training on new contexts when small amounts of data are available. The reduction in computational cost is achieved by processing the radar sensed data of gestures in the form of time maps, to minimize the input data size. This approach enables the adaptation of simple convolutional neural network (CNN) to new hand poses, thus easing the integration of the model into a hardware-constrained platform. Moreover, the use of a Variational Autoencoders (VAE) to reduce the gestures' dimensionality leads to a model size decrease of an order of magnitude and to half of the required adaptation time. The proposed framework, deployed on the Intel® Neural Compute Stick 2 (NCS 2), leads to an average accuracy of around 84% for unseen gestures when only one example per class is utilized at training time. The accuracy increases up to 92.6% and 94.2% when three and five samples per class are used.

INDEX TERMS Artificial neural networks, edge computing, FMCW, intel neural compute stick, knowledge transfer, meta learning, human computer interaction, radar, variational autoencoder.

I. INTRODUCTION

HCI represents a primary field of study to enable the communication between humans and systems [1]. A classic and widely used HCI method exploits the conductivity of a user's finger or skin touch with a capacitive surface [2], [3]. Although a precise technology, this approach requires direct contact with the user and may not be versatile in specific contexts [4]. In recent years, the development of technologies such as optic or radio-frequency has radically increased the interfacing capability in all application areas [5]. Many advances in the field focus on vision-based interfacing, i.e. the use of camera sensors such as Red Green Blue (RGB) and Time of Flight (ToF) [6]–[9]. In Fact, Camera sensors

The associate editor coordinating the review of this manuscript and approving it for publication was Cheng Hu.

bring the advantage of touchless communication. Nevertheless, Camera-based solutions lead to potential privacy issues and failures with poor light conditions in the environment. In comparison, radio-based methods are not directly affected by light and can also be used to estimate user actions through walls or barriers [10]. Wi-Fi-based systems can be robustly deployed in the HCI context even when the usage environment or the user orientation changes considerably [11]–[13]. Yet, Wi-Fi technology often requires the generation of high output power and a continuously running module to ensure operation. In contrast to this, radar technology, thanks to a more adaptable system power mode management, is finding increasing interest in the field of HCI applications [14]. Among the various radar modulation techniques, the Frequency Modulated Continuous Way (FMCW) is particularly suitable in the context of action recognition by providing simultaneously accurate information of the range and the velocity of targets [15], [16].

Among the various interfacing approaches, hand gesture represents a natural and easily interpretable communication mean [17], [18]. For this particular purpose, radars find wide use and can even be miniaturized and integrated into smartphones or other portable devices, such as the Google Soli [19]. State-of-the-art technology can allow hand movement sensing with high spatial resolution but must be coupled with an action recognition algorithm to enable HCI communication. Camera-based systems can find solutions based on computer vision techniques, such as skin color, skeleton, or motion recognition [20]. For radar applications, however, given the difficulty of recognizing the shape and contours of the hands, Deep Learning solutions are often adopted [21].

Machine learning finds applications in the most varied research areas, both for direct task solving and as a powerful computational tool for speeding up and modeling processes. Multiple topologies such as VGGNet [22], ResNet [23] and Inception [24] have been developed in the recent years to solve complex tasks with very high accuracy. Such networks, however, to be trained and adapted, require a fair amount of computing power and resources, which is not suitable for deployment on most edge devices [25]. Appropriate models for edge devices require specific topologies and learning processes, often leading to a trade-off between performance and adaptability. Research in the edge domain focuses mainly on two areas, namely, the topologies optimization for deployment and post-training adaptation [26]. Effective methods for reducing the size of models and the computation parameters include the use of information compression methods such as SqueezeNet [27] and depth-wise separable filters like the MobileNets [28]. Post-training model optimization can instead be achieved without important loss of performance, by employing techniques like quantization [29], factorization [30], distillation [31] and pruning [32]. Edge efficient models development has recently led to an industry movement toward such a framework. Indeed, devices with embedded deep learning components account for a large portion of state-of-the-art HCI and Internet of Things (IoT) solutions [33]. In most of today's industrial applications of deep learning, however, models and related learning algorithms are tailor-made for specific tasks [34], [35]. While application-tuned models can achieve outstanding performance in complex and multidimensional problems, they also imply visible adaptability and interpretability weaknesses [36], [37]. The target algorithms often employ a lot of data to achieve high and robust performance. In addition, data labeling can be expensive because it may require experts, or might be sparse and depending on real-time applications [38].

A relatively new branch of machine learning, called Meta-Learning [39] has emerged to find proper solutions to problems where the adaptability on few data is essential. The idea behind Meta-Learning is to use contextual information, so-called meta-knowledge, to build a more robust model, easily adaptable to new tasks with little data. A specific subclass of meta-models called optimization-based [40] allows the transfer of meta-information between tasks via gradient method or parameters averaging. The general optimization-based approach is to learn, for a set of tasks, the best possible initialization of the parameters of a model, to make it easily adaptable in new contexts. The optimization is usually performed in the form of an episodic adaptation within two iterative steps. In base learning (inner-loop), a model learns how to solve an N-ways task, where N is the number of classes randomly sampled from the large set of training classes (if classification). In the outer loop, called meta-learning, an algorithm adapts the model following a generalization learning objective. The examples (shots) used in the inner loop are called of *support*, while the data used with the objective of generalization are called of *query*. While many meta-learning techniques rely on complex topologies and forms of gradient transmission to achieve high-performance [41]–[43], optimization-based techniques, given their generality, can enable the deployment of optimized models on current edge technologies.

In this paper, we propose a meta-learning optimization-based approach that enables the fast model adaptation on new gestures also at the edge. Radar-based gesture recognition in short-range applications (in the range of a few cm) represents a potential method of communication or interfacing with portable systems such as smartphones. Depending on the desired application, fast adaptation to new gestures or data may be essential. This approach can be useful not only for recognizing new action types, but also for adapting to individuals with motor disabilities or visual impairment, who are unable to perform an action in a conventional way.

We first design a radar-based setup and preprocessing suitable for the meta-learning context. Using the sensor *BGT60TR13C* FMCW [44] we gather data for a total of twenty hand gestures, performed by five users in three different environments. The collected raw data follow a definite frequency-based preprocessing and are then elaborated on

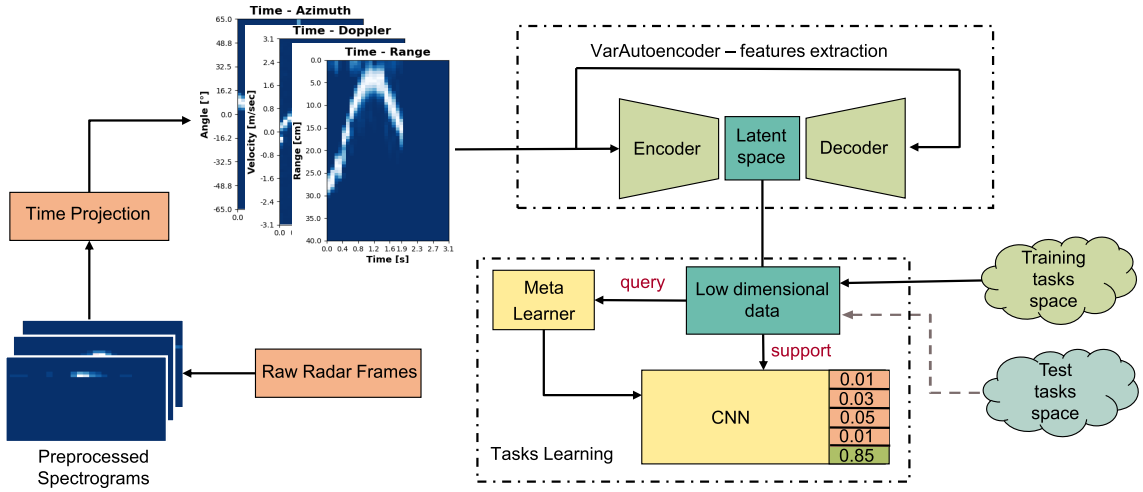

FIGURE 1. Block Diagram of the proposed model. For each gesture, the sequence of raw radar frames is initially processed in frequency. It is then elaborated and concatenated in the time domain to obtain the range, velocity, and azimuth angle of arrival information of the targets. A VAE, pre-trained on 12 training gesture classes, compresses the three-channel image into a constrained multivariate latent distribution of dimension 15. The meta-algorithm training is done on a sequence of randomly sampled tasks, exploiting the support and query data in an N-ways K-shots approach. As the meta-iterations progress, the adaptability performance is assessed on tasks sampled from the 8 test classes.

the time axis for dimensionality reduction without relevant information loss. Then, employing Model Agnostic Meta-Learning (MAML) [45] as the base algorithm, we introduce some methods to increase the generalization capabilities of the model over new tasks. Respectively we introduce dynamic metaclass weighting (DMCW), task-specific gradient clipping (TSGC), and evaluation-based Gaussian noise summation (EGNS). We then describe how, by using part of a pre-trained Convolutional Variational Autoencoder (Conv-VAE) in the classifier, we can greatly reduce the size of the meta-model without a major loss in generalization performance. The block diagram of the proposed approach is depicted in Fig. 1.

We then compare the achieved results with other state-of-the-art meta-learning algorithms, showing how our solution leads to an optimal trade-off between network size, accuracy, and latency time. Finally, we perform an offline adaptation of the base model on Raspberry® Pi4 with Intel® Neural Compute Stick 2 (NCS 2), to enable the embedded application and the fine-tuning on eight defined test gestures. In this context, the training time required to tune the model to a new task on Raspberry® Pi4 and the inference time per single prediction on NCS 2 are provided. The main contributions of this paper are as follows:

1) Implementation of a proof-of-concept user-definable radar-based hand gesture recognition system at the edge. To the best of our knowledge, the first implementation at the edge in the field of radar-based user-definable gesture recognition.
2) Use of a specific preprocessing aiming at simplifying both time domain dependency and computational complexity.
3) Conceptualization of some techniques aimed at increasing the generalization capability of the algorithm on unseen gestures.
4) Design of a dimensionality reduction method, through a Conv-VAE, suitable for the optimization-based meta-learning at the edge.

II. RELATED WORKS

In this section, we first analyze both general and radar-based methods for hand gesture recognition. We then focus on the specific works that involve the use of little training data, such as meta-Llarning.

A large part of the literature focuses on the use of vision-based techniques for gesture recognition [46]. Sagayam and Hemanth [47] proposed a method for interpreting and classifying RGB Camera-based hand gestures using a 1-D hidden Markov Model (1-D HMM). Instead of complex dynamic programming methods, a heuristic method called Artificial Bee Colony (ABC) is used for the 1-D HMM optimization. The presented algorithm leads to accurate and fast models compared to other state-based methods. The state-based approach, however, can be too slow and unsuitable for adaptation in new contexts. De Smedt *et al.* [48] presented a method for classifying dynamic hand skeletal data using the linear Support Vector Machine (SVM). Kinematic descriptors of gestures are extracted from the input data and then statistically and temporally coded. The pre-segmented data are then fed to the SVM for recognition. The method leads to a very low computational latency in all experiments and great performance on various datasets, but it is highly dependent on the time encoding. Liao *et al.* [49] illustrated a system for hand gesture-based alphabet recognition using both RGB and depth information. The Hough transform

applied to the depth information is used to remove the background from the color images. The feature extraction is done through a Double-Channel Convolution Neural Network (DC-CNN). The method achieves robust performance on a large dataset but, the multi-channel approach makes it unsuitable for recognition based on other classes of sensors. Tran *et al.* [50] proposed a method that uses an RGB-D camera and a 3D Convolution Neural Network (3DCNN) ensemble to accurately and robustly recognize both gestures and fingertip position in real-time. Recognition is achieved through the hand skeleton-joint extracted by the recordings' in-depth information. The model leads to a satisfactory accuracy of 97.12% on the test data. Despite the accuracy, the method is computationally expensive and complex to adapt to new gestures, such as those featuring finger-tip oscillations. Azad *et al.* [51] presented a method for classifying sequences of hand depth maps by analyzing and sampling temporal information at various levels. Gesture features in the form of spatiotemporal information are derived using Weighted Depth Motion Maps (WDMM). The extracted information is further reduced by Principal Component Analysis (PCA) and classified by a single hidden layer feed-forward neural network (SLFN) with an Extreme Learning Machine (ELM). Their proposed method achieves satisfactory results in three different datasets, outperforming the results obtained by deep learning methods. Although this algorithm is less computationally complex than most deep models, its architecture is also closely related to the nature of the data and difficult to generalize to other types of input.

Other classes of sensors used for touchless gesture recognition solutions involve ultrasonic sensors and Wi-Fi technology. Das *et al.* [52] explored the use of ultrasonic sensors for gesture recognition as low power and low-cost alternative to optical sensors. The classification is achieved by combining a CNN and a Long Short-Term Memory (LSTM) for both spatial and temporal feature extraction. Ultrasonic sensors can represent an alternative approach to radars but, if compared to the latter, can be subject to interference phenomena and not always application-adaptable. Zheng *et al.* [53] presented a system for gesture recognition via Wi-Fi that enables adaptability in various domains (i.e. orientation of people, locations, and environments). The method exhibits zero-effort cross-domain adaptability employing a domain-independent body-coordinate velocity profile (BVP) estimation method. A Deep Neural Network (DNN) trained on a set of BVPs thus allows for robust recognition of as many as 15 hand gestures across domains without re-training needs. Despite the versatility of the approach, the method still requires 5,000 samples for training and is not easily adaptable to new types of gestures.

The literature on recognition using radar sensors mainly focuses on Doppler or FMCW modulated radars.

Skaria *et al.* [54] illustrated a method for classifying 14 types of gestures captured by a Doppler radar via deep CNN. The radar device employed is a miniaturized, low-cost dual-channel receiver model. To successfully differentiate among Doppler radar sensed gestures, the phase difference between the two antennas is exploited to infer the angle of arrival (AoA). The method shows a classification accuracy of 95% on the test and a clear differentiation between classes. However, Doppler radars, due to their limitation in spatial resolution, find limited use for gesture recognition commonly employed for HCI. Lee *et al.* [55] presented a method to improve the prediction accuracy in hand gesture recognition by *BGT60TR13C* FMCW using deep learning. The algorithm uses domain adaptation to address the problem of gesture misrecognition due to performance differences as users vary. The information extracted from the FMCW radar is frequency processed to obtain Range-Doppler Maps (RDMs). A 3D-CNN with an Inception structure processes the spatio-temporal sequence of the RDMs for classification. In parallel, an adversarial domain discriminator is used to minimize the differences between gestures performed by different users. With this method, the accuracy of 98.8% is achieved on seven gestures performed by ten users. The domain adaptation represents a powerful generalization tool in the presence of few data but requires a related source domain rich in labels to succeed. Chmurski *et al.* [56] depicted how a neural network with depthwise separable convolutions can lead to high accuracy values for FMCW radar-based gesture recognition while operating in a low-power and resource-constrained environment. The model, built on eight hand gestures is optimized and deployed on the Coral Edge TPU Board. This approach although efficient is hardly adaptable to new actions.

In recent years, HCI research is evolving towards the adaptability of systems in new contexts and with little data. Rahimian *et al.* [57] presented a class of few-shot learning architectures for gesture recognition via electromyography. The designed approach succeeds in the generalization with only a few examples per gesture by combining temporal convolution with an attention mechanism using a meta-learning approach. The contextual information acquired with experience allows the model to adapt quickly even to new gestures which have never been observed in the training phase. Lu *et al.* [58] illustrated a one-shot method for gesture recognition using 3D-CNN, by exploiting transfer learning methodology from models trained with big datasets to strengthen the one-shot predictor. This approach, tested on several Vision benchmark datasets, leads to good classification and latency results. Madapana *et al.* [59] explored Hard Zero-Shot Learning (HZSL) on vision-based datasets for dynamic gesture recognition. The work tries to solve the classification problem by exploiting only limited training information in the form of semantic description. Although the achieved performance is far from direct data classification, this paper shows that even minimal information can lead a model to learn how to generalize.

Some work focused directly on the use of self-learning techniques for radar-based gestures. Fan *et al.* [60] have shown how a meta-learning approach can bring high generalization benefits for radar-based gesture recognition using

FMCW modulation. The information obtained by radar for a set of seven gestures is preprocessed in the form of time maps to extract the information of range, velocity, and angle of arrival of the hands. The data is then fed in the form of tasks to an LGM-Net-based architecture [61]. The method leads to an accuracy of 97.3% on the 2-ways task employing 5 test samples per class. However, the multi-branch structure and the elaborate learning process make it computationally complex. Zent *et al.* [62] have recently presented a work that focuses on gesture recognition using a Doppler sensor. The information is processed as micro-Doppler spectrograms to map over time the change in frequency caused by the hand displacement atop the sensor. Rather than learning a direct mapping between gestures and labels, the presented method, called Weighting Network, based on Relation Networks [41], learns to compare the test spectrograms with those used for training. The presented solution has the great benefits of not requiring adaptation training for new gesture types and a relatively small number of parameters. However, the architecture needs inherently to learn the direct relationship between the *support* and *query* examples in the comparison module. This characteristic, intrinsic to Relation Net-based models, can lead as exposed in [63] to lack of adaptation in the testing phase compared to other methods. Further, in [64], it has been shown how an optimization-based method can be effectively employed for HCI via FMCW radar by exploiting simplified interfacing based on hand gesture sequences and a classical CNN for classification.

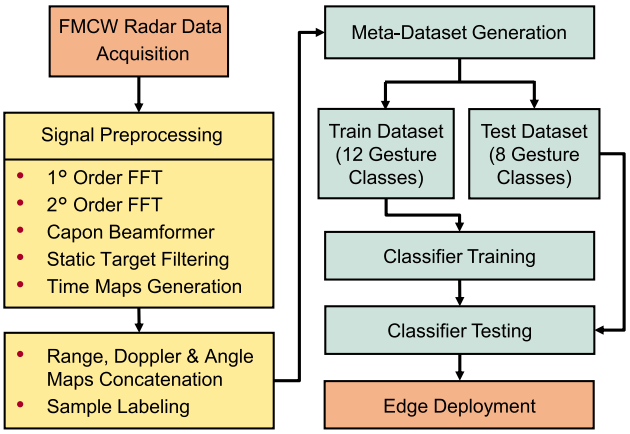

FIGURE 2. Data acquisition through FMCW radar, signal preprocessing, meta-dataset generation, and training and testing process for the proposed meta-learning-based hand gesture classifier. The orange-colored parts are hardware related. In yellow is the data processing, while in green is the classifier part. The frequency analysis is enabled by Fast Fourier Transform (FFT).

FIGURE 3. *BGT60TR13* Radar System. The radar sensor, is mounted on top of the board.

III. SYSTEM DESCRIPTION AND RADAR PREPROCESSING

In this section, we present the various components of the system (i.e., hardware details, operating parameters, and recording setup) and the proposed preprocessing of the data collected via radar.

A. GENERAL OVERVIEW OF THE PROPOSED FRAMEWORK

The proposed framework is shown in Fig. 2. First of all, the raw radar signals are preprocessed to extract both frequency and time information. The data obtained for each gesture in the shape of range, Doppler, and AoA temporal maps, are then used as meta-dataset for the optimization-based meta-learning approach. Twelve types of gestures are used to train the classifier, whereas the other eight are utilized for testing. After the training process, the model is deployed through the Raspberry® Pi4 on the NCS 2 and, adapted on new test gestures to exhibit the proof-of-concept for adaptability.

B. RADAR BOARD

In this work, gesture sensing is performed by the *BGT60TR13C* FMCW radar sensor [44], manufactured by Infineon Technologies AG. The sensor is equipped with a Transmit (TX) and tree Receive (RX) channels with an included antenna integrated into the package. The information is processed channel-wise in several steps, through the board to which the sensor is connected Fig. 3. The operating principle of the sensor relies on linear frequency modulation of continuous waves. The TX transmits periodic signals called chirps and, the RXs receive signals reflected from the targets located in front of the sensor. During operations, the instant local oscillations are mixed with the reflected signals and result in an output signal called the Intermediate Frequency (IF). The IF signal is then passed to a baseband chain and digitalized through an analog-to-digital converter (ADC) with 12-bit resolution.

The *BGT60TR13C* is a miniaturized solution with a center frequency f_0 of 60 GHz and a bandwidth of about 6 GHz that enables a high range solution ($\approx$ 2 cm). The phase analysis of the IF signal, exploiting the micro-Doppler effect [65], can also enable the discrimination of displacements with millimeter accuracy. Thanks to the 3 RX channels orthogonal to each other, the radar enables the estimation of both azimuth (between 65 and -65 degrees) and elevation (between 45 and -45 degrees) AoA of targets. This system also features power mode management and an operation-optimized duty cycle to reduce power consumption to only 5 mW for applications within the 5 m range. The *BGT60TR13C* represents so, a low-power and miniaturized solution for short-range sensing applications.

C. RADAR PARAMETERS CONFIGURATION

The *BGT60TR13* system allows to transmit for each so-called radar frame, a sequence of N_c chirps with a single signal

TABLE 1. Radar sensor parameters configuration.

Symbol	Quantity	Value
f_0	center frequency	60 GHz
B_w	bandwidth	[57 – 63] → 6 GHz
F_s	sampling frequency ADC	2 MHz
N_c	number of chirps	64
t_c	chirp time duration	390.4 µs
n_s	samples per chirp	32
$Azi.AoA$	azimuth angle of arrival	-65 – 65 deg
fps	frames per second	10

duration time t_c along the slow-time dimension. Each chirp also consists of a number n_s of samples along the fast-time dimension. The transmitted signals use the saw-tooth wave function modulation to enable a linear behavior during the chirp rise phase. For an FMCW radar, the range resolution Δr and the maximum detection range R_{max} can be derived through the following formulas:

$$\Delta r = \frac{c}{2B_w} \tag{1}$$

$$R_{max} = \frac{\Delta r}{2} \cdot n_s \tag{2}$$

where c is the speed of light and B_w represents the frequency bandwidth around the central f_0 frequency. A bandwidth of 6 GHz, between 57 GHz and 63 GHz, has been chosen to enable a high range resolution of about 2.5 cm. The number of samples per chirp has been set to 32 for enabling the detection of targets up to a range of 40 cm. Further, an ADC sampling frequency F_s of 2 MHz has been chosen not to limit R_{max} because of signal conversion. The velocity resolution Δv and the maximum detectable velocity in a given direction V_{max} can be computed as:

$$V_{max} = \frac{c}{4f_0 t_c} \tag{3}$$

$$\Delta v = \frac{V_{max} \cdot 2}{N_c}. \tag{4}$$

A number of 64 chirps per frame N_c with single signal duration time t_c of 390.4 μs, has been chosen to allow a V_{max} of about 3.14 m/s and a Δv of about 9.8 cm/s respectively. The parameters used for radar configuration in the hand gesture sensing application are in Table 1.

D. RADAR SIGNAL PREPROCESSING

The raw sensed radar data are not easily interpretable due to spatial resolution constraints and the influence of noise and environment surrounding the targets. While it may be possible to develop an application based on raw data as input, this would involve the training on a large amount of data that only partially contains the target information. In this work, we propose to process the signals first in frequency to extract and separate the shifts in range and velocity caused by the hands located in front of the sensor. For each detected gesture, the information is processed frame-wise and then concatenated in time to project the range and velocity contents in the 2-D plane. In such a way, Range Time Maps (RTM) and Doppler Time Maps (DTM) are generated. Exploiting the signal sensed by two RX channels, the AoA azimuth is also estimated via Capon beamformer algorithm [66]. The azimuth information is then processed and projected on the temporal plane for each frame to form Angle Time Maps (ATM).

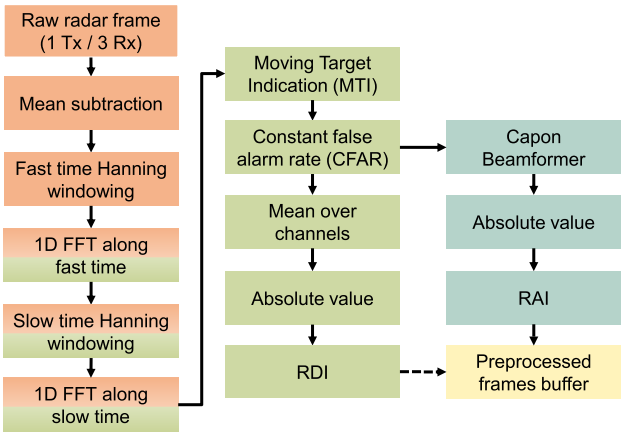

FIGURE 4. Diagram illustrating step by step the preprocessing used on each radar frame. In orange are shown the operations performed in the time domain, in green those done in the frequency domain, in blue the AoA computation.

1) SINGLE FRAME PREPROCESSING

For this application, the IF signal $S_{IF}(n)$ for each of the three available RX channels $n \in N_{RX}$ is employed to build a frame. For each n channel, the data are arranged in a 2-D matrix with slow time for the x-axis (raws) and fast time for the y-axis (columns). For each frame, by frequency analysis using Fast Fourier Transform (FFT), the Range-Doppler Image (RDI) is first calculated. The AoA azimuth is then estimated using the Capon algorithm to build the Range Angle Image (RAI). Fig. 4 depicts the various preprocessing steps used to obtain frame-wise RDI and RAI.

The first part of the preprocessing consists of the following steps.

1) First the mean values, computed over the fast time are subtracted along the slow time axis.
2) The data are then multiplied with a Hanning window along the fast time to minimize spectral leakage effects for frequency analysis.
3) The 1-D FFT along fast time is executed to extract the range information.
4) Hanning windowing is applied along the slow time axis.
5) The 1-D FFT along slow time is performed on data to extract the velocity information.
6) The moving target indication (MTI) is next applied to discriminate targets against unwanted background information, aka clutter (5).

$$S_{IF}(n) = \alpha \cdot S_{IF}(n) + (1 - \alpha) \cdot \overline{S_{IF}}(n) \tag{5}$$

where α is a parameter in the range [0 – 1] set to 0.9, and $\overline{S_{IF}}(n)$ the updated moving average for each frame.

7) A Constant False Alarm Rate (CFAR) algorithm is used for each channel n to filter the frequency peaks and increase the Signal-to-Noise Ratio (SNR).

To further increase the SNR for the RDI computation, the absolute value of the average of $S_{IF}(n)$ over the N_{RX}, as shown in (6).

$$RDI = \left| \frac{1}{N_{RX}} \cdot \sum_{n=0}^{N_{RX}} S_{IF}(n) \right| \tag{6}$$

After using CFAR, the $S_{IF}(n)$ associated with the two RX channels placed in the horizontal plane is processed by Capon beamforming for the AoA computation. The absolute value is then calculated and the RAI is generated.

2) GESTURE SENSING AND TIME PROJECTION

Gesture sensing begins when an average $\overline{S_{IF}}$ for the three RX channels is higher than a defined threshold, which is computed every time the sensor is turned on for a new recording session (i.e. new environment or new user). The threshold is determined as the average value of the last 20 collected frames (2 s) and it is used for comparison at every timestamp during operation. A gesture is considered gathered when the threshold is not exceeded for 5 consecutive frames. The recording window has a length of 3.1 s and therefore contains up to 32 frames for every performed action.

The stored frames, are then preprocessed in the form of RDI and RAI and mapped into a lower-dimensional space to compute the RTM, DTM and ATM. For each RDI or RAI, belonging to a sequence of matrices definable as $A : \{1, \ldots, m\} \times \{1, \ldots, m\} \times \{1, \ldots, t\} \rightarrow \mathbb{R}$, where $t \geq 1$, the goal is to find the index (x, y) corresponding to the maximum value $a_{x,y}^{max}$.

$$A_{m,n} = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \ldots & a_{mn} \end{bmatrix}$$

where $m \times n$ represents the range and Doppler dimensions for the RDI and range and angle dimensions for the RAI. The information, corresponding to the distance and velocity of the target from the sensor, is extracted by taking from the RDI the $Col_x(A)$ and the $Row_y(A)$ respectively. The AoA azimuth is instead extracted from the RAI by taking the $Row_y(A)$. The concatenation of the obtained rows or columns for the whole gesture duration leads to the generation of the RTM, DTM, and ATM. Fig. 5 illustrates graphically the principle of range information extraction given a sequence of frames. Each hand pose is represented by 3-channel information (RTM, DTM, and ATM). The gestures collected with fewer than 32 frames are expanded via zero padding at the end of the time sequences. All instances are normalized channel-wise in the [0 – 1] range.

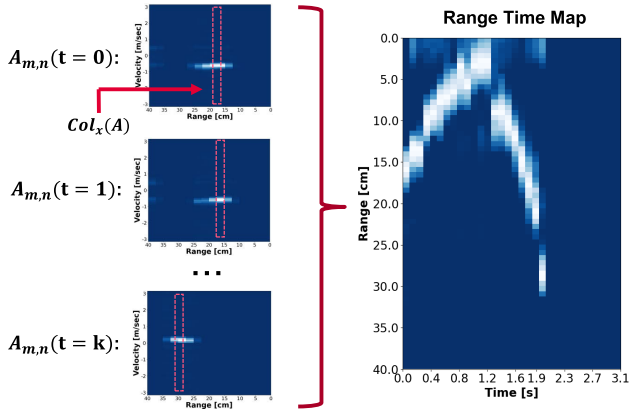

FIGURE 5. Example of time projection for an RTM generation.

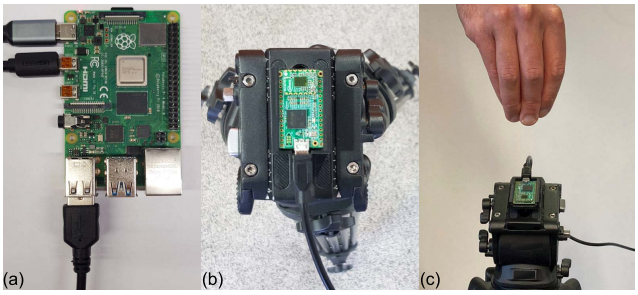

FIGURE 6. Recording setup for gestures sensing. (a) shows the Raspberry® Pi4 employed for data recording. (b) depicts the BGT60TR13 radar board on the tripod. (c) shows an example of performed action for the class "rubbing."

FIGURE 7. Recording setup for the offline proof-of-concept of the system generalization capability at the edge. The Raspberry® Pi4 is used for data preprocessing, model adaptation and script running. The NCS 2 enables the deployment of the developed meta-learning model for a specific setup.

E. RECORDING SETUP

In this work, we sensed gestures via radar for a total of twenty classes. The recording setup for data collection consists of a Raspberry® Pi4 and the *BGT60TR13* board. The radar board is mounted on a tripod through a 3D-printed case. With the defined radar configuration, a maximum detection range of 40 cm implies the potential use as short-range application only. Such setup is therefore meant for handheld or turnstile gesture recognition interfaces. The setup in its components is depicted in Fig. 6. The actions have been performed by a total of five users and in three different environments (office,

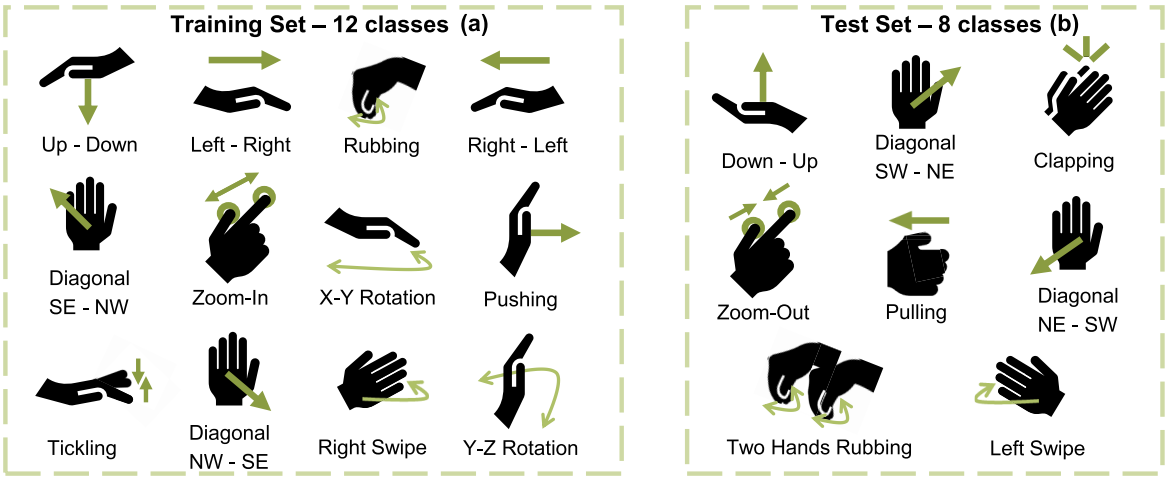

FIGURE 8. Gestures vocabulary for the meta-training (a) and meta-test (b) datasets. N, S, W and E represent the cardinal points.

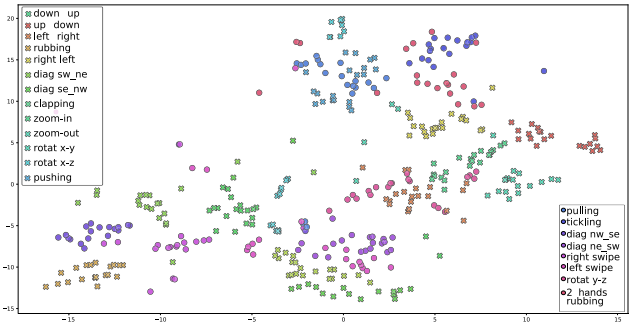

FIGURE 9. 2-D components t-SNE representation of the twenty gestures of the dataset. Classes belonging to $D^{m-train}$ are represented with a cross marker. Classes belonging to D^{m-test} are represented by a point marker.

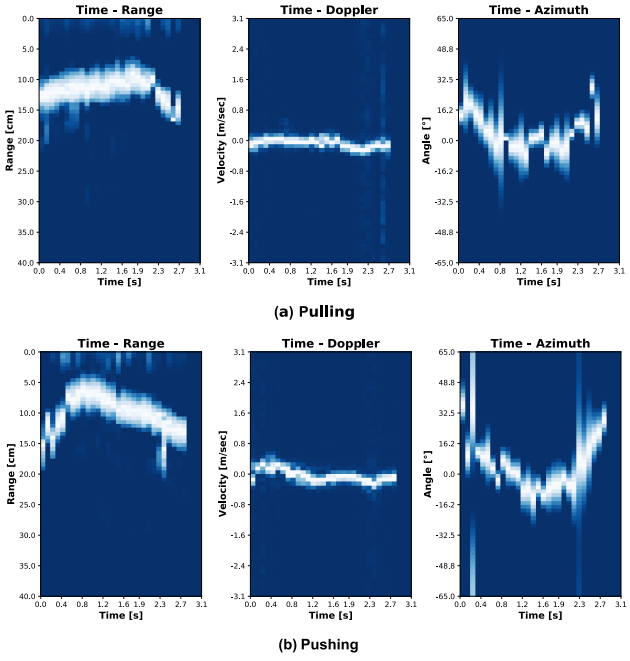

FIGURE 10. Comparison of RTM, DTM and, ATM between (a) Pulling, and (b) Pushing. In this example, the range information allows a clear distinction between the two classes.

hall, and outdoor). These specific environments have been chosen among several possible, as they represent three contrasting application contexts. In the office, the presence of static furniture and devices placed in the radar's field of view can result in added reflections and subsequent noise in the preprocessed signals. The hall and outdoors instead, represent two wide environments where, in first approximation, only the arm and potentially the body of the subject performing the gesture fall within the field of view of the radar. The data were collected partly outdoors to avoid possible dependencies from secondary reflections given by devices and metal ducts placed in the hall environment. The consent has been obtained from users prior to data collection and as much anonymity and privacy as possible were maintained during the data collection and processing phases. Individuals of varying height [1.60 – 1.85] m and age [25 – 40] years with no relevant motor or visual impairments have been engaged in the experiment. The only information given to the users before performing the gestures were the radar orientation, and the maximum duration of the gestures of 3.1 s. The data have not been saved in online archives and/or published. The chosen gestures are those most commonly employed for HCI in touchless applications.

Only in the final test phase, to demonstrate the offline proof-of-concept of the system's adaptability to new gestures, the developed model is deployed on Raspberry® Pi4 and NCS 2. This setup is shown in Fig. 7.

F. GESTURES DATASET

For the meta-learning approach, the gestures are split by classes between a meta-training $\mathcal{D}^{m-train}$ and meta-test $\mathcal{D}^{m-test}$ sets. Fig. 8 illustrates the twelve training and eight test gestures, respectively. The division of gestures has

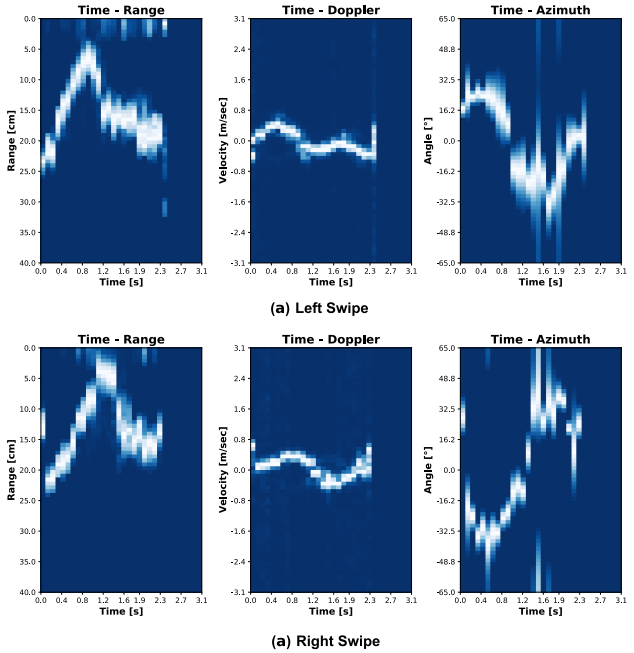

FIGURE 11. Comparison of RTM, DTM and, ATM between (a) left swipe, and (b) right swipe. In this example, the azimuth information allows a clear distinction between the two classes.

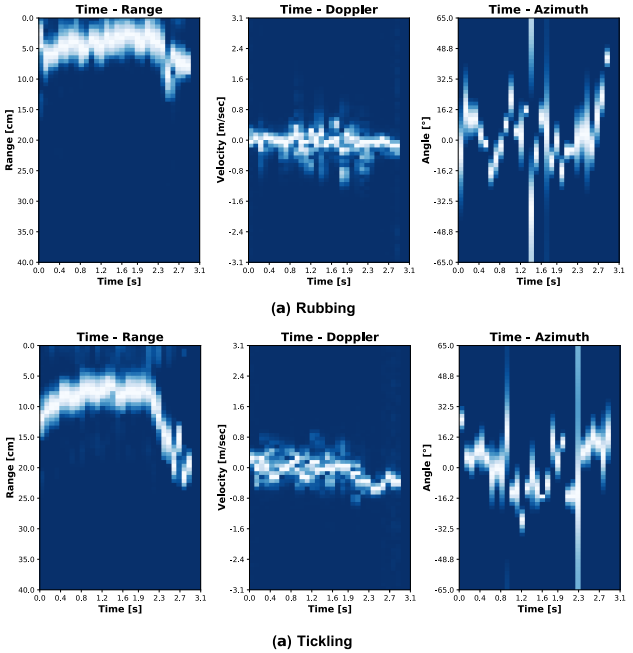

FIGURE 12. Comparison of RTM, DTM and, ATM between (a) rubbing and (b) tickling. Local oscillation caused by finger movement in the velocity profile can be noted for both classes.

been performed randomly, with the only constraint to keep, in the two datasets, the sets of gestures that are opposite to each other. A t-distributed Stochastic Neighbor Embedding (t-SNE) representation of the gestures in two components is shown in Fig. 9.

Extracting both range and azimuth information is crucial for correctly distinguishing some gestures from others. Examples where RTM and ATM clearly allow a distinction between two classes are shown in Fig. 10 and Fig. 11, respectively. Velocity information can improve the separation between classes, especially concerning the spatial plane in which the gestures are performed. In addition, such information can help distinguish actions characterized by local finger oscillations, such as rubbing and tickling Fig. 12.

IV. PROPOSED METHOD

In this section, we propose our approach, which belongs to the class of optimization-based meta-learning algorithms. We first introduce some methods to increase the model's generalization capability in comparison to the state-of-the-art. We then present the adopted CNN topology and the benefits of using a pre-trained Conv-VAE as a backbone in the meta-learning phase to reduce the number of parameters.

A. OPTIMIZATION-BASED META-LEARNING

In a conventional optimization-based meta-learning approach for deep learning, the optimization consists of two iterative steps performed over the distribution of tasks $p(\mathcal{T})$, to train a model represented by a parametric function f_θ with parameters θ. The two optimization steps are the following:

1) In *base-learning*, for a batch of N tasks, an inner learning model $f_{\theta'_n}$ with parameters θ'_n, tries to solve each task $\mathcal{T}_n$, given a dataset $\mathcal{D}_{\mathcal{T}n}$ and a task related loss function to minimize $\mathcal{L}_{\mathcal{T}n}(f_\theta)$.
2) In *meta-learning*, an outer algorithm makes use of the information obtained through back-propagation of the gradient in the inner learning phase to update the internal algorithm. The model trained during base learning also minimizes an outer loss function $\mathcal{L}_{ext}(f_{\theta'_n})$.

If the loss function defined for the task is differentiable, the internal optimization is often performed by Stochastic Gradient Descent (SGD) in K batches of training examples belonging to $\mathcal{D}_{\mathcal{T}n}$. The θ' parameters are computed as:

$$\theta'_n = \theta - \gamma \cdot \sum_{k=1}^{K} \nabla_\theta \mathcal{L}^{(k)}_{\mathcal{T}n}(f_\theta) \tag{7}$$

where γ is the inner loop learning rate of the meta-algorithm. In [45], Finn *et al.* present a very general method called Model Agnostic Meta-Learning (MAML) where the meta-optimization across tasks is also performed via SGD, by minimizing the function $f_{\theta'_n}$ with respect to θ, for each single task or N tasks sampled from $p(\mathcal{T})$.

$$\begin{aligned} &\min_\theta \frac{1}{N} \cdot \sum_{n=1}^{N} \mathcal{L}^{(n)}_{ext}(f_{\theta'_n}) \\ &= \frac{1}{N} \cdot \sum_{n=1}^{N} \mathcal{L}^{(n)}_{ext}(f_{\theta - \gamma \cdot \sum_{k=1}^{K} \nabla_\theta \mathcal{L}^{(k)}_{\mathcal{T}n}(f_\theta)}) \end{aligned} \tag{8}$$

$$\theta \leftarrow \theta - \beta \cdot \frac{1}{N} \cdot \nabla_\theta \sum_{n=1}^{N} \mathcal{L}_{ext}^{(n)}(f_{\theta_n'}) \tag{9}$$

where β in (9) is the outer loop learning rate. In MAML for few-shot supervised learning, two different data sets are defined for each task $\mathcal{T}_n$. Support samples $\mathcal{D}_n$ for base learning and query $\mathcal{D}_n'$ for the inter-tasks generalization step in the meta-learning phase. As can be seen in (8), meta-gradient involves a gradient through a gradient and can lead to instability during training as well as resulting computationally expensive. Antoniou *et al.* [67] present various modifications to the MAML to enhance the learning stability and also the generalization capability.

In our work, we adopt MAML as the base algorithm, with a task batch size N of 1 and, we exploit some of the methods presented in [67] to improve the training stability. Specifically, we leverage the following contributions:

- **Multi-Step Loss Optimization (MSL)**: instead of minimizing the outer loss function after the completion of all base learning steps for support set task $\mathcal{D}_n$, we do an update after each inner-epoch $i \in I$, composed of K batches, using $\mathcal{D}_n'$. Specifically, we exploit a set of importance weights v_i that enables a higher loss contribution for the latest i in I.

$$\theta \leftarrow \theta - \beta \cdot \nabla_\theta \sum_{i=1}^{I} v_i \sum_{k=1}^{K} \mathcal{L}_{ext}^{(k)}(f_{\theta_k'}) \tag{10}$$

  In addition, as the meta-iterations performed on the distribution of tasks $p(\mathcal{T})$ progress, the relative weights of early epochs are decreased and, those of the late epochs are increased. This strengthens the ability to learn from every individual $\mathcal{T}_n$ task without potentially destabilizing learning. In comparison to the method proposed in [67], where the update of the outer loss is performed after each step towards the support set task, we suggest an update after each inner-epoch. This leads to a trade-off between intra-task learning steps and computational complexity.
- **Derivative-Order Annealing (DA)**: the use of the second-order gradient involves some computational expenses and can make the optimizer inefficient and unstable during the early training phase of MAML. To overcome these problems, we anneal the derivative order in the first 50 meta-iterations by exploiting the first-order gradient information only.
- **Cosine Annealing of Meta-Optimizer Learning Rate (CA)**: to fine-tune the optimization via the outer algorithm as the meta-iterations progress, we apply a cosine annealing scheduling on the optimizer. This yields an increase in generalization performance without impacting the per task computation $\mathcal{T}_n$.

We besides propose some methods that can increase the generalization capability of MAML without bringing any increase in computational complexity in evaluation and testing. Respectively, for this purpose, we present the Dynamic Meta Class Weighting (DMCW), Task-Specific Gradient Clipping (TSGC), and the Evaluation-based Gaussian Noise Summation (EGNS).

1) DYNAMIC META CLASS WEIGHTING

In a task learning approach with only a few data, a model can easily overfit the training instances leading to weak classification performance on the testing instances. Few examples per class may not be informative enough for the description and lead to significant misclassifications in testing. One way to counter this is to use in the inner loop, for each task $\mathcal{T}_n$, a set of class weights $\forall\, c \in C$, where C represents the number of ways. Specifically, we propose to compute after each inner-epoch, for each $c \in C$, a weight v_c which is inversely proportional to the number of correct predictions. The idea is to sample for each task $\mathcal{T}_n$, a balanced set of examples $\mathcal{D}_c \neq \{\mathcal{D}_n; \mathcal{D}_n'\}$ on which each inner epoch performance can be dynamically evaluated. For a given class c, with corresponding M weighting examples x_m, the normalized weight $\overline{v}_c$ in the range [0–1] is computed as follows:

$$\overline{v}_c = \frac{1}{\sum_{c=1}^{C} v_c} \cdot \sum_{m=1}^{M} (\hat{y}_m - y_m) \tag{11}$$

where y_m represents each instance-associated label, $\hat{y}_m$ the predicted label after every inner-epoch and, v_c the computed weights before normalization. The resulting $\overline{v}_c$ weights are used both in the *base learning* and the *meta-learning* updates after each batch k in K. Respectively:

$$\theta_n' = \theta - \gamma \cdot \sum_{k=1}^{K} \overline{v}_c^{(k)} \cdot \nabla_\theta \mathcal{L}_{\mathcal{T}n}^{(k)}(f_\theta) \tag{12}$$

and for the *meta-learning* update, through MSL:

$$\theta \leftarrow \theta - \beta \cdot \nabla_\theta \sum_{i=1}^{I} v_i \sum_{k=1}^{K} \overline{v}_c^{(k)} \cdot \mathcal{L}_{ext}^{(k)}(f_{\theta_k'}) \tag{13}$$

Each inner update improves intra-task classification performance by bringing more attention to minimizing $\mathcal{L}_{\mathcal{T}n}$ on classes whose examples have been poorly classified. In addition, the outer update allows inter-task propagation of the information obtained with the weights $\overline{v}_c$ to improve generalization performance.

2) TASK-SPECIFIC GRADIENT CLIPPING

Task training performed with little data for a given number of epochs I brings benefits in some cases but can also lead to gradient explosion and instability in others. The model can so overfit on a given task, making generalization to others less effective. One solution to this is performing gradient clipping for the intra-task updates when the gradient exceeds a threshold, as presented by Pascanu *et al.* [68]. In our case, we suggest using clipping in the intra-task phase, for each batch k in K on the when the gradient g computed for $\mathcal{L}_{\mathcal{T}n}$

exceeds a certain threshold h:

$$\begin{cases} g = \dfrac{\partial \mathcal{L}_{\mathcal{T}n}(f_\theta)}{\partial \theta}, \\ g \leftarrow \dfrac{g \cdot h}{\|g\|}, \qquad if \ \|g\| > h \end{cases} \tag{14}$$

where $\|g\|$ represents the L2 norm computed on the gradients. We propose further not to use gradient clipping for the intra-task update on queries via $\mathcal{L}_{ext}$. By doing so, the query update grants a higher contribution to the whole optimization-based procedure.

3) EVALUATION-BASED GAUSSIAN NOISE SUMMATION

Training on a sequence of tasks for a large number of meta-iterations can make the algorithm too specific on $\mathcal{D}^{m-train}$ and thus decreasing the generalization capability on $\mathcal{D}^{m-test}$ leading to the so-called meta-overfitting. One way to counteract such behavior on $D^{m-train}$ is to increase the complexity of the task when the performance becomes very high. One way to make a task n more complex is to add Gaussian noise to the examples x_n in $\mathcal{D}_n$ or to their embedded representations as to the output of the hidden layers of the model. Specifically, we propose to sum to the output of various depths of the model, random Gaussian noise in the interval $[-\sigma; \sigma]$ from the distribution $\mathcal{N}(\mu, \sigma^2)$ generated for each batch k in K. This Gaussian noise is activated for a new training task only when the validation accuracy, performed on a sequence of tasks, sampled by $\mathcal{D}^{m-train}$, exceeds a defined threshold.

B. PROPOSED TOPOLOGIES

For the optimization-based meta-learning approach, we propose the use of two topologies. First a traditional one, consisting of sets of convolutional layers for features extraction. Then, a structure that uses part of a Conv-VAE as a backbone to considerably reduce the number of parameters in the overall topology. For both neural networks, the goal is, given a task $\mathcal{T}_n$, to map the sequence of RTMs, DTMs, and ATMs belonging to a gesture to the respective class.

1) CONVOLUTIONAL NEURAL NETWORK

The first topology consists of three convolutional layers with the final dense layer. The convolutional layers use 128, 256, and 512 filters respectively, with a kernel size 3×3 and a stride of 2. Each of these layers is followed by batch normalization, to increase the training stability for each batch k, and by the ReLU activation function. A Flatten layer and a Dense layer are attached to the last of the three convolution blocks. The Dense layer output neurons correspond to the number of classes in the experiment. The classification is enabled through the Softmax activation function, which maps the output vector into a classes probability distribution. The topology is depicted in Fig. 13.

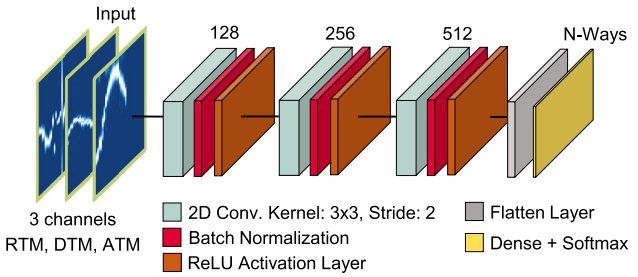

FIGURE 13. CNN topology. For each gesture, consisting of RTM, DTM, and ATM information in-depth channels, features are extracted from three blocks of convolutional layers. A final dense layer with Softmax activation enables the classification. The number of filters per convolution is noted above the respective blocks.

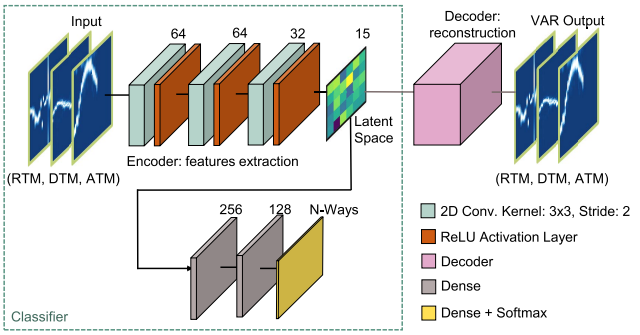

FIGURE 14. Conv-VAE and Dense topology. To significantly reduce the number of parameters compared to the convolutional model, the classification is done by exploiting the encoder of a Conv-VAE pre-trained on $\mathcal{D}^{m-train}$. For the categorical classification, three Dense layers connected to the final layer of the encoder (latent space) are used. The number of filters and neurons in the various layers is noted above the respective blocks.

2) Conv-VAE AND DENSE

The second topology exploits part of a Conv-VAE, pre-trained on $\mathcal{D}^{m-train}$, to significantly squeeze the input size. The Conv-VAE compresses the three-channel information (RTM, DTM, and ATM) into a constrained multivariate latent distribution of dimension 15. The Encoder part of the Conv-VAE model is then extracted and concatenated to a sequence of Dense layers for task training. The Dense layers consist of 256, 128, and N output neurons respectively, corresponding to the number of ways for the experiment. Also for this topology, the outputs of the last layer are mapped in a classes probability distribution through Softmax. The layers extracted from the Conv-VAE are also trained during optimization-based meta-learning for the N-ways classification objective. The topology is shown in Fig. 14.

V. EXPERIMENTAL SETUP

In this section, we present and analyze the performed optimization-based meta-learning experiments. Specifically, we conducted 1-shot 2-ways, 1-shot 5-ways, 3-shots 5-ways, and 5-shots 5-ways experiments. The algorithm and methods presented are mainly analyzed in the 5-ways setup, to depict their advantages. The algorithm has been developed in the

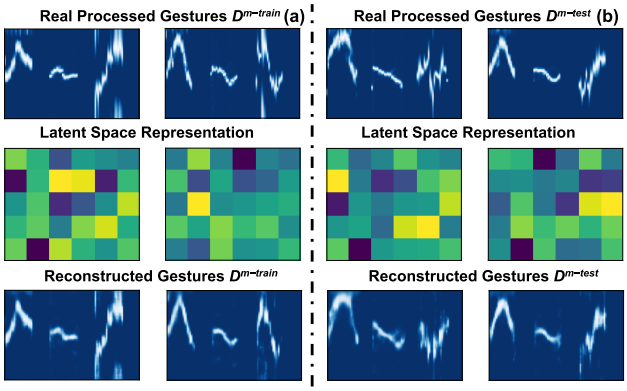

FIGURE 15. Example of latent space generation (heatmap representation) and reconstruction using Conv-VAE, for train (a) and test (b). For better visualization of the instances, the RTM, DTM, and ATM channels are concatenated as a single image.

Python programming language through the TensorFlow® module. The performance tests for the state-of-the-art comparison, have been performed on a eight generation Intel® Core™ i5 processor (4-cores). At the edge side, the Raspberry® Pi4 and NCS 2 have been employed. Consequently, the RaspbianOS operating system has been utilized. To run the model on NCS 2 and optimize the inference process, we used the OpenVino module on Python.

A. META-LEARNING EXPERIMENTS

All experiments have been performed in a similar setup for the two topologies (CNN and Conv-VAE + Dense). For the topology with the Conv-VAE, the network on the $\mathcal{D}^{m-train}$ dataset is first pre-trained. The employed loss function and optimizer are binary crossentropy and Adam respectively. A learning rate of 1e-4 is used for Adam. The training is conducted on 200 epochs with a latent dimension of 15, i.e., 30 descriptive parameters of the set of multivariate Gaussian distributions. Since Conv-VAE is part of the category of deep generative networks, it can also partially reconstruct $\mathcal{D}^{m-test}$ instances without further training. An example of reconstruction on sampled classes from both $\mathcal{D}^{m-train}$ and $\mathcal{D}^{m-test}$ is displayed in Fig. 15.

For the 5 ways experiments, the task training is performed through 4 inner epochs and an inner-batch of size 2 for 1-shot, and size 3 otherwise. For both *base-learning* and *meta-learning* phases, the Adam optimizer is used with β_1 and β_2 equal to 0 and 0.5 respectively. The inner learning rate is set to 8e-4, whereas the meta-learning rate has an initial value of 7e-4 with a decay step of 2,000. The chosen number of meta-iterations is 2,200, while the classes for each task are randomly sampled by $\mathcal{D}^{m-train}$. The loss function chosen for the classification is categorical crossentropy. In the evaluation phase, accuracy statistics are saved and processed every 220 iterations in the shape of box plots. For experiments with the EGNS, a task buffer of length 5 has been chosen, with a $\mathcal{D}^{m-train}$ validation accuracy threshold of 89%, 95% and 98% for 1-shot, 3-shots, and 5-shots, respectively. For the TSGC experiments, the gradient is clipped when the L2 norm exceeds 0.5. For the DMCW, a total of 10 samples per class is used for the computation of the weights. The generated models are finally tested on 1,000 tasks sampled by $\mathcal{D}^{m-test}$. For DMCW and EGNS, the final task training is performed as a traditional single-task optimization approach. For TSGC, gradient clipping is also executed on the training batches. The EGNS and DMCW are exclusively used during meta-iterations, to increase the model's generalization capability over one or a few new examples of unseen classes. The achieved prediction accuracy, model size, adaptation time, and latency are evaluated and compared with state-of-the-art techniques. In both the evaluation and testing phases, 10 examples per class are used for testing. This means that in the 5-ways experiments, 50 test examples per task are utilized.

B. PERFORMANCE EVALUATION

We first present the results obtained on a single experiment, showing the benefits achievable on unseen classes thanks to an optimization-based meta-learning approach. Then, we conduct an ablation study, by analyzing the contributions of the individual proposed methods, for both proposed topologies. Next, we compare our achieved results with those of some existing techniques in terms of neural network size, prediction accuracy, and latency. All the experiments for the proposed methods and ablation study have been performed on a 4-core eight generation Intel® Core™ i5 processor. Regarding adaptation at the Edge, we display the results of adaptation time to new tasks and model deployment on Raspberry® Pi4 and NCS 2.

1) EXPERIMENT ANALYSIS

The metric used to evaluate the training performance of each model is validation accuracy. This parameter is estimated after each meta-iteration, by evaluating the model on new sampled tasks. For each validation two tasks are sampled by the $\mathcal{D}^{m-train}$ and $\mathcal{D}^{m-test}$ respectively. A box-plot of task statistics is built every 220 meta-iterations. Generalization ability can be assessed by observing the variation in the box plots as meta-iterations progress. In a successful experiment, we observe the increase of the median accuracy on the sequence of box plots, as well as the reduction of the intervals of percentiles and whiskers. The trend of box plots for the experiment with EGNS for the CNN topology is shown in Fig. 16. The contribution of EGNS is combined with the basic MAML + MSL + DA + CA algorithm, which we term $^{+}$MAML. Another possible way of assessing the generalization capability is to observe the distribution of validation accuracy as meta-iterations increase. Usually, for the first training tasks, the accuracy tends to assume a multimodal shape due to different complexity in tasks resolution. In the training time, the model learns to resolve better new tasks thanks to the improved parameters' initialization. This leads the accuracy distribution to have a negatively skewed tendency towards the 100% correct classification. The accuracy

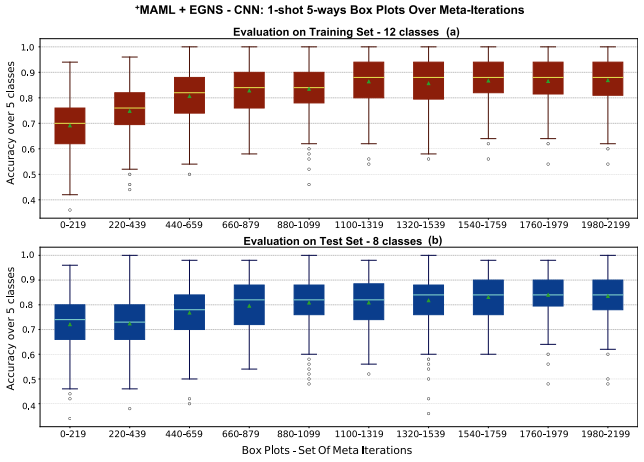

FIGURE 16. The trend of box plots generated on classification accuracy in the validation phase for the EGNS experiment with CNN topology. In red (a), are the box plots built on the tasks sampled from the meta-train dataset, while in blue (b) are those built over the meta-test. The mean and median values are represented for each box plot by a triangle and a line, respectively.

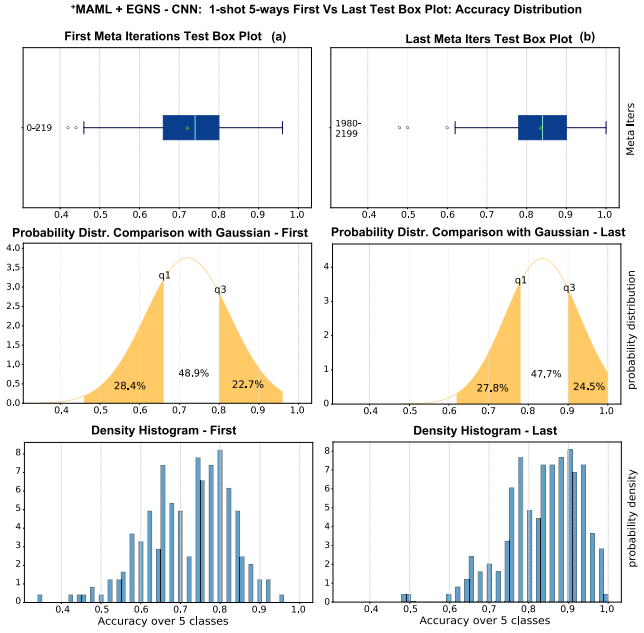

FIGURE 17. Density histogram of validation accuracy on test for the EGNS experiment with CNN topology. First (a) and last (b) meta iterations. Values q1 and q3 on the Gaussian indicate first and third quartiles, respectively. Percentages indicate the amount of data in the sections of the distribution. The accuracy, which does not assume a Gaussian distribution, exhibits a negative skew for the last 220 meta-iterations.

density histograms, generated for the first and last 220 meta-iteration box-plots, are shown in Fig. 17 for the CNN - EGNS experiment. The quartiles and range percentages are noted in the middle plot on a Gaussian distribution that could be associated with the box plot. Roughly by definition, 50% of the values are contained between the first and third quartiles of the box plots. The actual accuracy distribution, however, as can be seen, does not assume a Gaussian shape.

TABLE 2. CNN topology. Average accuracy results of 5-ways experiments with 95% confidence intervals, computed over 1,000 final test tasks of $\mathcal{D}^{m-test}$. Individual methods are implemented in each experiment to the base algorithm.

Accuracy 5-ways	1-shot [%]	3-shots [%]	5-shots [%]
Base ($^{+}$MAML)	82.85 ± 0.55	92.07 ± 0.31	$\mathbf{94.19 \pm 0.26}$
DMCW	82.32 ± 0.56	90.80 ± 0.40	93.64 ± 0.27
EGNS	$\mathbf{84.36 \pm 0.55}$	$\mathbf{92.54 \pm 0.30}$	93.87 ± 0.25
TSGC	$\mathbf{84.28 \pm 0.56}$	$\mathbf{92.85 \pm 0.30}$	94.16 ± 0.26
DMCW+EGNS	82.53 ± 0.57	91.10 ± 0.34	93.85 ± 0.25
DMCW+TSGC	82.81 ± 0.53	91.39 ± 0.34	93.49 ± 0.26
EGNS+TSGC	83.97 ± 0.54	92.02 ± 0.32	94.15 ± 0.25
All	83.29 ± 0.55	91.90 ± 0.31	93.92 ± 0.26

TABLE 3. Conv-VAE+Dense topology. Average accuracy results of 5-ways experiments with 95% confidence intervals, computed over 1,000 final test tasks of $\mathcal{D}^{m-test}$. Individual methods are implemented in each experiment to the base algorithm.

Accuracy 5-ways	1-shot [%]	3-shots [%]	5-shots [%]
Base ($^{+}$MAML)	76.31 ± 0.63	87.18 ± 0.38	90.78 ± 0.31
DMCW	76.86 ± 0.61	88.95 ± 0.38	92.06 ± 0.30
EGNS	78.69 ± 0.61	87.99 ± 0.37	90.90 ± 0.29
TSGC	78.67 ± 0.60	88.24 ± 0.35	91.18 ± 0.28
DMCW+EGNS	78.67 ± 0.57	89.46 ± 0.37	92.59 ± 0.27
DMCW+TSGC	$\mathbf{80.09 \pm 0.59}$	$\mathbf{90.59 \pm 0.33}$	$\mathbf{92.97 \pm 0.27}$
EGNS+TSGC	$\mathbf{80.25 \pm 0.58}$	88.73 ± 0.34	92.07 ± 0.28
All	79.19 ± 0.57	89.93 ± 0.33	92.59 ± 0.26

The generalization outcome can even be observed on the individual classes by generating a cumulative confusion matrix for sets of meta-iterations. In Fig. 18 are depicted the confusion matrices of the first and last 550 meta-iterations for the EGNS with CNN topology experiment. As can be noticed from the matrices, as the iterations progress, the model learns to solve quicker new tasks thanks to the updated initialization. This also applies to the unseen classes belonging to $\mathcal{D}^{m-test}$.

Some actions are more complex to distinguish between each other because of similarities in patterns, thus leading to specific prediction errors. It can be noted, for example, the misclassification between right swipe and diagonal nw-se in the confusion matrix on $\mathcal{D}^{m-train}$ and specularly that between left-swipe and diagonal nw-sw for $\mathcal{D}^{m-test}$.

2) RESULTS ANALYSIS

All the experiments have been performed for both the proposed topologies, analyzing the combination of the presented methods against the base algorithm $^{+}$MAML. Each experiment, tested on 1,000 final test tasks, has been repeated three times. The average accuracy results for the 5-way experiments are presented in Table 2 and Table 3, respectively.

For the CNN topology experiments, the total number of trainable parameters in the model is 1,562,629. This large number of parameters, as can be noticed through accuracy results in Table 2, allows the model to generalize well, guaranteeing with fast-adaptation, excellent results on unseen classes. For such a topology, the DMCW method brings no performance benefit. The model size enables extracting more features from each data while query update after each epoch through MSL reduces the possibility of overfitting. The

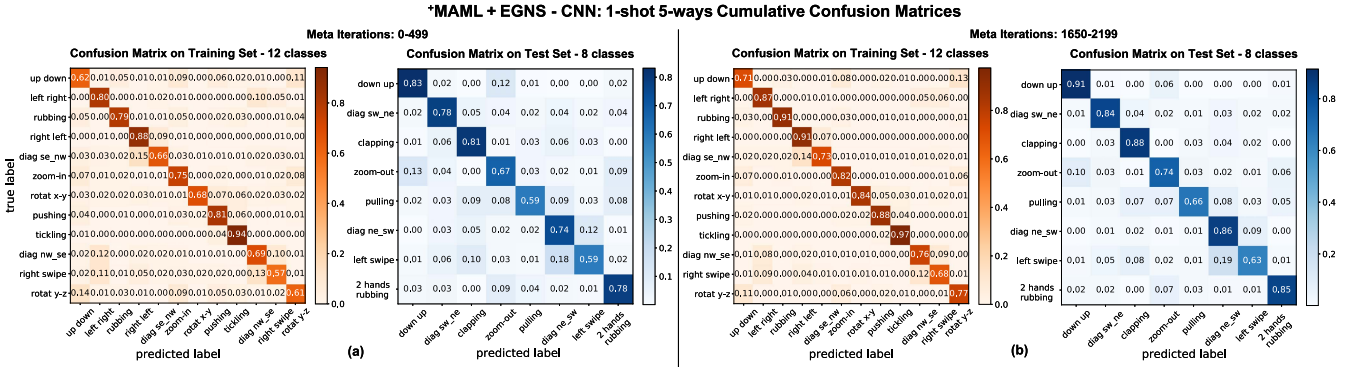

FIGURE 18. Cumulative confusion matrices for the EGNS experiment with the CNN topology. Confusion matrices are obtained on the first (a) and last (b) 550 meta-iterations in the validation phase for both training and test classes.

DMCW is even more counterproductive as the number of shots increases. In such a case, the model comprehends better the differences among classes, thanks to the higher number of examples. For 1 and 3 shots experiments, individual use of EGNS and TSGC leads to the highest accuracy. These techniques give less importance to single tasks, thus favoring the meta-learning objective. In the 5-shot approach, the $^{+}$MAML algorithm, allows without additional contributions, to achieve the highest accuracy. The information provided by the training instances is then enough to compensate for possible overfitting and exploding gradients. For the first topology, none of the experiments where the contributions are combined bring accuracy benefits. For a model with high feature extraction capability, the use of both EGNS and DMCW techniques can make each task locally complex and misleading. Thus, decreasing the significance of the meta-learning updates.

For simulations with the Conv-VAE+Dense topology, the total number of trainable parameters in the model drops to only 118,851. Due to input information mapping to small size, individual tasks may be more affected by overfitting phenomena. In this case, the DMCW introduces benefits compared to the basic version of the algorithm. This contribution is also beneficial with 3 and 5 shots, probably supporting the classification of the compressed information squeezed by the backbone. For this topology, the best results are achieved by combining the DMCW and TSGC methods. The classification of low-dimensional representations is aided by class weighting for individual tasks. The TSGC, on the other hand, avoids exploding gradient and gives more importance to the outer loop update at the end of each inner epoch. The combination of the three methods brings equal or less satisfactory results than combining two of them for 1 and 3 shot experiments. With the use of 5 shots, the single techniques contributions do not lead to better results than with $^{+}$MAML. This is probably due to the higher amount of data available, which leads to smoother task training. The accuracy results for both topologies in the 1-shot 2-ways approach are presented in Table 4.

TABLE 4. Average accuracy results of 1-shot 2-ways experiments with 95% confidence intervals, computed over 1,000 final test tasks of $\mathcal{D}^{m-test}$. Individual methods are applied in each experiment to the base algorithm, for both topologies.

Accuracy 1-shot 2-ways	CNN [%]	Conv-VAE+Dense [%]
Base ($^{+}$MAML)	90.57 ± 0.73	88.89 ± 0.77
DMCW	90.78 ± 0.71	86.71 ± 0.98
EGNS	91.17 ± 0.73	87.62 ± 0.84
TSGC	$\mathbf{92.70 \pm 0.67}$	$\mathbf{91.23 \pm 0.69}$
DMCW+EGNS	91.53 ± 0.71	86.51 ± 1.00
DMCW+TSGC	91.38 ± 0.73	90.43 ± 0.74
EGNS+TSGC	$\mathbf{92.88 \pm 0.64}$	90.86 ± 0.69
All	$\mathbf{93.05 \pm 0.63}$	87.32 ± 0.86

TABLE 5. Training times to adapt to new tasks for both topologies on the 4-core Intel® CPU. Times, given a number of ways and shots, are calculated as the average of the adaptation time of all experiments, each tested and averaged over 1,000 final test tasks of $\mathcal{D}^{m-test}$.

Adaptation Time	CNN [ms]	Conv-VAE+Dense [ms]
1-shot 2-ways	498	206
1-shot 5-ways	1,180	647
3-shots 5-ways	2,548	1,254
5-shots 5-ways	4,278	1,991

For 1-shot 2-ways experiments, the greatest benefits are achieved through TSGC for both the topologies. For a 2-ways application, the DMCW contributions are counterproductive or not significant. Class weighting with only two categories can easily skew the learning towards one of them, especially with small input sizes as for the second topology. For similar reasons, the model can learn to over-depend on noise augmented inputs via EGNS and rank worse on the test data. For CNN, the use of combined EGNS and TSGC brings some benefits, mainly preventing overfitting in the base-learning phase, given the higher simplicity of the tasks. The accuracy reached with the three techniques combined depicts how preventing over-dependence on the individual tasks can favor the generalization aim.

The average adaptation times to new tasks on the 4-core Intel® CPU for the two topologies are listed in Table 5.

As can be seen from the table, the model size of the Conv-VAE topology, which is an order of magnitude smaller than

TABLE 6. Best-in-class results compared to the state of the art. Average accuracy results of the experiments with 95% confidence intervals, computed over 1,000 final test tasks of $\mathcal{D}^{m-test}$. The various algorithms have been tested under similar evaluation conditions on the 20 gestures dataset.

Accuracy	5-ways [%]			2-ways [%]
	1-shot	3-shot	5-shot	1-shot
Reptile	63.95 ± 0.62	86.82 ± 0.39	89.68 ± 0.40	87.27 ± 0.84
MAML (2nd Order)	78.95 ± 0.59	90.71 ± 0.36	93.41 ± 0.23	88.53 ± 0.78
LGM-Net	77.57 ± 0.20	84.38 ± 0.14	89.98 ± 0.10	85.04 ± 0.17
Weighting Net	81.17 ± 0.48	**93.47 ± 0.30**	**94.47 ± 0.28**	**95.61 ± 0.43**
Proposed (CNN)	**84.36 ± 0.55**	92.54 ± 0.30	94.19 ± 0.26	93.05 ± 0.63
Proposed (Conv-VAE+Dense)	80.25 ± 0.58	90.59 ± 0.33	92.97 ± 0.27	91.23 ± 0.69

the CNN, allows a reduction of the adaptation time by half for the 1-shot experiments. The time required to adapt to a new task is further reduced for Conv-VAE when more than 1 example per class is employed. Regardless of the method used, the inference time on CPU to predict the class of a single example is on average 64 ms for both topologies in the 5-ways approach.

3) COMPARISON WITH EXISTING TECHNIQUES

The best-achieved results, obtained through the various experiments and topologies, are compared with both meta-learning state-of-the-art and classical optimization-based algorithms, trained on $\mathcal{D}^{m-train}$ and tested on $\mathcal{D}^{m-test}$. Respectively, the Reptile [69] and MAML algorithms for the optimization-based class and Weighting Net and LGM-Net, employed in the papers [61] and [62] are trained on our proposed gestures dataset. For the comparison, similar evaluation conditions are used. The Reptile and MAML (2nd Order) algorithms are utilized to train the proposed CNN topology for 2,200 meta-iterations. The topology presented in [61], adapted to 3-channel gesture information, has been employed for the LGM-Net. The Weighting Net, with a feature dimension of 64, has been adapted to the shape of the gestures and, the relative embedding module has been trained to extract features only from the *support* instances. The accuracy results for the state-of-the-art algorithms, averaged over three repetitions, are presented in Table 6.

As can be noticed from Table 6, the proposed method with CNN topology performs the best in the 1-shot 5-ways experiment, leading to better results than the Weighting Net by around 3 %. In all the other experiments, the proposed method performs slightly less accurately only compared to the Weighting Net. With more than one shot, the Weighting Net has the advantage of being able to mediate the predictions obtained thanks to a sequence of comparisons of the *query* image with those of *support*. However, with the availability of only one example per class, it lacks this great feature and loses robustness. The proposed methods though, lead in all the experiments to better results than all the other optimization-based methods. For simple experiments (2-ways) or a higher number of shots, the difference in accuracy obtained between the methods gets narrower. In such conditions, even the simplest algorithms can achieve high feature extraction from samples. So, the resolution of the tasks becomes less dependent on the initialization making the employed generalization techniques less effective.

TABLE 7. Best-in-class results compared to the state of the art. Number of trainable parameters per topology and experiment, computed over 1,000 final test tasks of $\mathcal{D}^{m-test}$. The various algorithms have been tested under similar evaluation conditions.

Model Size	5-ways	2-ways
Reptile	1,562,629	1,513,474
MAML (2nd Order)	1,562,629	1,513,474
LGM-Net	300,421	298,882
Weighting Net	229,157	226,034
Proposed (CNN)	1,562,629	1,513,474
Proposed (Conv-VAE+Dense)	**118,851**	**118,464**

The comparison in terms of model size is presented in Table 7.

In terms of the number of parameters, the Conv-VAE+Dense approach enables the generation of an order of magnitude smaller models compared to the CNN. Even if in terms of accuracy the second topology performs a few percentage points worse than the Weighting Net, it requires about half as many parameters for tasks resolution. Furthermore, among the compared methods, the Conv-VAE topology results in the one with the least number of required variables.

Table 8 presents the time required for adaptation to a new task (Ta) and the single-sample inference (Ti) for the considered algorithms. Reptile and MAML are tested using the same methodology as the proposed optimization-based models. As they are utilized on the CNN topology, they lead to results very similar to those of the proposed methods and are, therefore, excluded from this table. For LGM-Net, the two values of Ta and Ti are summed, given the high degree of interdependence between the modules (Embedding, MetaNet, and TargetNet) in its structure. For the Weighting Net, Ta is estimated as the required time to map the *support* examples to a reduced size via the EmbeddingNet. In this case, Ti is computed as the needed time to process and classify a *query* example through the entire model pipeline after the *support* adaptation. In terms of adaptation time (Ta), the proposed models take longer than the Weighting Net. On the other hand, the optimization-based models enable the instance classification in a significantly short time (Ti) and in a way that is independent of the number of training shots. In the 5-shot experiment, the proposed topologies require only a quarter of the time needed by the Weighting Net

TABLE 8. Best-in-class results compared to the state of the art. Adaptation time (Ta) and latency of prediction on single sample (Ti) per topology and experiment, computed over 1,000 final test tasks of $\mathcal{D}^{m-test}$. The various algorithms have been tested under similar evaluation conditions on the 4-core Intel® CPU.

Adaptation (Ta) + single prediction time (Ti)	5-ways [ms]			2-ways [ms]
	1-shot	3-shot	5-shot	1-shot
	Ta + Ti	Ta + Ti	Ta + Ti	Ta + Ti
LGM-Net	1,710	4,310	6,290	1,340
Weighting Net	**92** + 143	**278** + 215	**476** + 287	**39** + **55**
Proposed (CNN)	1,180 + **64**	2,548 + **64**	4,278 + **64**	498 + 61
Proposed (Conv-VAE+Dense)	647 + 64	1,254 + 64	1,991 + 64	206 + 61

TABLE 9. Training times to adapt to new tasks for both topologies on Raspberry® Pi4 (without NCS 2). Times, given a number of ways and shots, are calculated as the average of the adaptation time of all experiments, each tested and averaged over 10 final test tasks of $\mathcal{D}^{m-test}$.

Adaptation Time	CNN [ms]	Conv-VAE+Dense [ms]
1-shot 2-ways	3,655	983
1-shot 5-ways	7,976	2,572
3-shots 5-ways	22,188	5,851
5-shots 5-ways	37,310	9,416

TABLE 10. Training times to adapt to new tasks for both topologies on Raspberry® Pi4 plus deployment time on NCS 2. Times, given a number of ways and shots, are calculated as the average of the adaptation time of all experiments, each tested and averaged over 10 final test tasks of $\mathcal{D}^{m-test}$.

Adaptation Time	CNN [ms]	Conv-VAE+Dense [ms]
1-shot 2-ways	3,678	2,265
1-shot 5-ways	7,416	3,544
3-shots 5-ways	13,283	4,562
5-shots 5-ways	21,567	6,701

for prediction. This brings a huge advantage in real-time applications or implementations at the edge.

4) EDGE IMPLEMENTATION

The topologies presented in this paper use only NCS 2 compatible layers and procedures. All models, pre-trained with the optimization-based approach on the 4-cores CPU, are adapted at the edge to single tasks generated by $\mathcal{D}^{m-test}$ via Raspberry® Pi4. The models are first tested on the Raspberry® Pi4 without connecting NCS 2, consequently using only the 4 ARM cores, to estimate adaptation time and inference on single sample. The computed task adaptation time are presented in Table 9. The models are then deployed on the NCS 2 and, prediction inference for each test sample is conducted at the device level. For the various experiments, the achieved results in terms of summation of task adaptation time on Raspberry® Pi4 and deployment on NCS 2 are presented in Table 10.

As can be noticed from the Table 9 and Table 10, as the number of samples per class increases, the time to adapt to a new task rises significantly for the CNN topology, requiring up to more than 21 s for an adaptation and deployment on the NCS 2. On the contrary, the Conv-VAE+Dense, given the much smaller number of parameters, requires less than 7 s for a 5-shots task. Conv-VAE+Dense can therefore lead to a saving of up to about two-thirds of the time. The results from Table 9 highlight generally longer adaptation times for both topologies on the Raspberry®. This is mainly due to computation limits, especially for the CNN topology as the number of shots increases. In addition, the models, once deployed on the NCS 2, allow much shorter single inference times (Ti) and therefore enable potential real-time applications with very low latency. The needed time for a single prediction after model adaptation is topology-dependent. For both 2-ways and 5-ways experiments, the model on NCS 2 requires an average of 5 ms and 4 ms for CNN and Conv-VAE+Dense, respectively. These values are significantly lower than those obtained only via Raspberry® Pi4 (Table 9), where the prediction of a single example takes on average 351 and 333 ms for CNN and Conv-VAE+Dense, respectively. The demonstrated results underscore how deploying on the NCS 2 can be a very advantageous strategy when very low latency is required. Since the adaptation is performed offline, the single inference time does not consider the time required for gesture sampling (Ts) and preprocessing time (Tp). These times are dependent on the type of gesture performed, its intrinsic duration, and the number of recorded frames before applying zero padding. Table 11 presents the computed Ts and Tp

TABLE 11. Recording (Ts) and preprocessing (Tp) times computed for a random example belonging to each class of gestures. The time Tp is computed over an average of 10 preprocessing repetitions on Raspberry® Pi4.

Gesture	Ts [ms]	Tp [ms]
Down - Up	2,700	1,198
Up - Down	2,900	1,285
Left - Right	400	209
Rubbing	2,900	1,320
Right - Left	1,800	802
Diagonal SW-NE	1,900	884
Diagonal SW-NW	2,600	1,173
Clapping	1,700	786
Zoom-In	2,000	905
Zoom-Out	2,100	942
X-Y Rotation	3,100	1,370
Pushing	3,100	1,399
Pulling	2,500	1,116
Tickling	3,100	1,408
Diagonal NW-SE	1,600	719
Diagonal NW-SW	2,100	941
Right Swipe	1,600	721
Left Swipe	2,000	890
Y-Z Rotation	1,800	815
Two Hands Rubbing	2,600	1,166

times for one example per class of each of the 20 gestures. The Tp values are obtained over an average of 10 preprocessing repetitions of the same example performed on Raspberry® Pi4. Thus, the total (end-to-end) time consists of the sum of Ts+Tp+Ti.

VI. CONCLUSION

In this paper we present a complete pipeline based on hand gestures performed on an FMCW radar, to exhibit a proof-of-concept of user-adaptability for novel unseen hand poses. The system solution, based on data collected for twenty different types of gestures, from five users in three different environments, allows not only the extraction of useful features of performed actions but also a fast adaptation to new gestures. The pipeline is composed of a first preprocessing phase, then a meta-learning approach to generate the best possible model initialization, and an edge-suitable adaptation to new tasks and classes never faced in the training phase. The specific preprocessing employed, thanks to the combination of techniques both in the frequency and time domain, allows extracting the main information of the gestures only, thus significantly reducing the size of the raw data collected by radar. The information constructed for each gesture, in the form of 3 channels, represents the hand distance from the radar, the action velocity, and the azimuth angle of arrival. A meta-learning optimization-based approach, trained on twelve of the processed gestures, depicts how new never faced tasks can be more easily solved, thanks to the context information extracted in the training phase. Three techniques, aiming at increasing the generalization ability of the model in comparison to the state-of-the-art, are presented: dynamic meta-class weighting, task-specific gradient clipping, and evaluation-based Gaussian noise summation respectively. The introduced methods have the great advantage of improving the model's parameters initialization in the training phase without directly affecting the final adaptation setup on the eight test classes. This enables both a more versatile implementation at the edge and a very fast prediction on new samples, reducing remarkably the computation latency. Further, compared to other state-of-the-art techniques, the optimization-based approach doesn't involve the comparison of the query samples with the support ones in the test phase, thus, bringing to an additional time latency reduction. Two different topologies for task resolution are presented. A first topology based on a series of convolution layers consents feature extraction for each sample thanks to a large number of defined parameters. A second topology instead, employs the encoding part of a Conv-VAE as a backbone to efficiently extract features, thus greatly reducing the number of model parameters. For such a topology, a greater effect of the presented optimization techniques is visible, thanks to the various contributions that counteract the effects of overfitting, exploding gradient, and meta-overfitting. Thanks to these features, this topology enables the generation of models that perform very well in terms of accuracy but with half the variables required in comparison to state-of-the-art. Moreover, the results obtained at the edge optimistically show how these algorithms can be used for real-time applications, aiding the adaptation to new users, gestures, and situations. To the best of our knowledge, this is the first user-adaptable model implemented at the edge for radar-based HCI.

On the other hand, the generated models lead to an accuracy that is lower than the state-of-the-art in several experiments. Other meta-learning algorithms, based on the classification of relations among examples, have the inherent advantage of leading to more robust predictions. Future work will explore the application at the edge of relational algorithms and potential methods of reducing the model size without harming generalization capabilities. Experiments with a broader set of gestures and examples will also be conducted, examining the generalization ability of the models across various splits of users and environments. Adaptive interfacing, based on an approach such as the one presented in this paper may be exploitable for people with motor or visual impairments, due to their inability to perform classic actions in a conventional manner. Since individuals without disabilities have been considered in this current work, direct studies will be conducted on analyzing how much radar adaptive interfacing can be used and relied on in these particular use cases.

REFERENCES

[1] A. Dix, J. Finlay, G. D. Abowd, and R. Beale, *Human-Computer Interaction*. London, U.K.: Pearson, 2003.

[2] K. Ahuja, P. Streli, and C. Holz, "TouchPose: Hand pose prediction, depth estimation, and touch classification from capacitive images," in *Proc. 34th Annu. ACM Symp. User Interface Softw. Technol.*, Oct. 2021, pp. 997–1009.

[3] J. Kim, J. Park, H. Kim, and C. Lee, "HCI(human computer interaction) using multi-touch tabletop display," in *Proc. IEEE Pacific Rim Conf. Commun., Comput. Signal Process.*, Aug. 2007, pp. 391–394.

[4] S. Kaminani, "Human computer interaction issues with touch screen interfaces in the flight deck," in *Proc. IEEE/AIAA 30th Digit. Avionics Syst. Conf.*, Oct. 2011, p. 6B4.

[5] J. A. Jacko, *Human Computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications*. Boca Raton, FL, USA: CRC Press, 2012.

[6] P. Dondi, L. Lombardi, and M. Porta, "Human-computer interaction through time-of-flight and RGB cameras," in *Proc. Int. Conf. Image Anal. Process.*, Cham, Switzerland: Springer, 2011, pp. 89–98.

[7] Y. Wang and C. Jung, "Interaction-free hand segmentation using Kinect camera," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, p. 4593.

[8] Y. Duan, H. Deng, and F. Wang, "Depth camera in human-computer interaction: An overview," in *Proc. 5th Int. Conf. Intell. Netw. Intell. Syst.*, Nov. 2012, pp. 25–28.

[9] M. Bohme, M. Haker, T. Martinetz, and E. Barth, "A facial feature tracker for human-computer interaction based on 3D time-of-flight cameras," *Int. J. Intell. Syst. Technol. Appl.*, vol. 5, no. 3/4, p. 264, 2008.

[10] M. Zhao, T. Li, M. A. Alsheikh, Y. Tian, H. Zhao, A. Torralba, and D. Katabi, "Through-wall human pose estimation using radio signals," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7356–7365.

[11] Z. Fu, J. Xu, Z. Zhu, A. X. Liu, and X. Sun, "Writing in the air with WiFi signals for virtual reality devices," *IEEE Trans. Mobile Comput.*, vol. 18, no. 2, pp. 473–484, Feb. 2019.

[12] H. Yan, Y. Zhang, Y. Wang, and K. Xu, "WiAct: A passive WiFi-based human activity recognition system," *IEEE Sensors J.*, vol. 20, no. 1, pp. 296–305, Jan. 2020.

[13] L. Chen, X. Chen, L. Ni, Y. Peng, and D. Fang, "Human behavior recognition using Wi-Fi CSI: Challenges and opportunities," *IEEE Commun. Mag.*, vol. 55, no. 10, pp. 112–117, Oct. 2017.
[14] H.-S. Yeo and A. Quigley, "Radar sensing in human-computer interaction," *Interactions*, vol. 25, no. 1, pp. 70–73, Dec. 2017.
[15] B. Li, X. Yu, F. Li, and Q. Guo, "Deep learning based target activity recognition using FMCW radar," in *Proc. Int. Conf. Artif. Intell. Commun. Netw.*, Cham, Switzerland: Springer, 2020, pp. 484–490.
[16] P. Vaishnav and A. Santra, "Continuous human activity classification with unscented Kalman filter tracking using FMCW radar," *IEEE Sensors Lett.*, vol. 4, no. 5, pp. 1–4, Apr. 2020.
[17] A. Haria, A. Subramanian, N. Asokkumar, S. Poddar, and J. S. Nayak, "Hand gesture recognition for human computer interaction," *Proc. Comput. Sci.*, vol. 115, pp. 367–374, Jan. 2017.
[18] Y. Xu and Y. Dai, "Review of hand gesture recognition study and application," *Contemp. Eng. Sci.*, vol. 10, no. 8, pp. 375–384, 2017.
[19] J. Lien, N. Gillian, M. E. Karagozler, P. Amihood, C. Schwesig, E. Olson, H. Raja, and I. Poupyrev, "Soli: Ubiquitous gesture sensing with millimeter wave radar," *ACM Trans. Graph.*, vol. 35, no. 4, pp. 1–19, 2016.
[20] M. Oudah, A. Al-Naji, and J. Chahl, "Hand gesture recognition based on computer vision: A review of techniques," *J. Imag.*, vol. 6, no. 8, p. 73, 2020.
[21] S. Ahmed, K. D. Kallu, S. Ahmed, and S. H. Cho, "Hand gestures recognition using radar sensors for human-computer-interaction: A review," *Remote Sens.*, vol. 13, no. 3, p. 527, Feb. 2021.
[22] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. pattern Recognit.*, Jun. 2016, pp. 770–778.
[24] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Sep. 2015, pp. 1–9.
[25] M. P. Véstias, "Deep learning on edge: Challenges and trends," in *Smart Systems Design, Applications, and Challenges*. Hershey, PA, USA: IGI Global, 2020, pp. 23–42.
[26] M. Verhelst and B. Moons, "Embedded deep neural network processing: Algorithmic and processor techniques bring deep learning to IoT and edge devices," *IEEE Solid-State Circuits Mag.*, vol. 9, no. 4, pp. 55–65, Nov. 2017.
[27] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," 2016, *arXiv:1602.07360*.
[28] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.
[29] M. Nagel, M. Fournarakis, R. A. Amjad, Y. Bondarenko, M. van Baalen, and T. Blankevoort, "A white paper on neural network quantization," 2021, *arXiv:2106.08295*.
[30] S. Swaminathan, D. Garg, R. Kannan, and F. Andres, "Sparse low rank factorization for deep neural network compression," *Neurocomputing*, vol. 398, pp. 185–196, Jul. 2020.
[31] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.
[32] D. Blalock, J. J. G. Ortiz, J. Frankle, and J. Guttag, "What is the state of neural network pruning?" 2020, *arXiv:2003.03033*.
[33] H. Li, K. Ota, and M. Dong, "Learning IoT in edge: Deep learning for the Internet of Things with edge computing," *IEEE Netw.*, vol. 32, no. 1, pp. 96–101, Jan. 2018.
[34] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, and S. Dieleman, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.
[35] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 25. Stateline, NV, USA, Dec. 2012, pp. 1097–1105.
[36] G. Marcus, "Deep learning: A critical appraisal," 2018, *arXiv:1801.00631*.
[37] F. K. Došilović, M. Brčić, and N. Hlupić, "Explainable artificial intelligence: A survey," in *Proc. 41st Int. Conv. Inf. Commun. Technol., Electron. Microelectron. (MIPRO)*, May 2018, pp. 210–215.
[38] M. M. Masud, C. Woolam, J. Gao, L. Khan, J. Han, K. W. Hamlen, and N. C. Oza, "Facing the reality of data stream classification: Coping with scarcity of labeled data," *Knowl. Inf. Syst.*, vol. 33, no. 1, pp. 213–244, Oct. 2012.
[39] J. Vanschoren, "Meta-learning: A survey," 2018, *arXiv:1810.03548*.
[40] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-learning in neural networks: A survey," 2020, *arXiv:2004.05439*.
[41] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1199–1208.
[42] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "Meta-learning with memory-augmented neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1842–1850.
[43] S. Ravi and H. Larochelle, "Optimization as a model for fewshot learning," in *Proc. Int. Conf. Learn. Represent.*, 2017.
[44] S. Trotta, D. Weber, R. W. Jungmaier, A. Baheti, J. Lien, D. Noppeney, M. Tabesh, C. Rumpler, M. Aichner, S. Albel, and J. S. Bal, "Soli: A tiny device for a new human machine interface," in *IEEE ISSCC Dig. Tech. Papers*, vol. 64, Feb. 2021, pp. 42–44.
[45] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1126–1135.
[46] S. S. Rautaray and A. Agrawal, "Vision based hand gesture recognition for human computer interaction: A survey," *Artif. Intell. Rev.*, vol. 43, no. 1, pp. 1–54, Jan. 2012.
[47] K. M. Sagayam and D. J. Hemanth, "ABC algorithm based optimization of 1-D hidden Markov model for hand gesture recognition applications," *Comput. Ind.*, vol. 99, pp. 313–323, Aug. 2018.
[48] Q. De Smedt, H. Wannous, and J.-P. Vandeborre, "Heterogeneous hand gesture recognition using 3d dynamic skeletal data," *Comput. Vis. Image Understand.*, vol. 181, pp. 60–72, Apr. 2019.
[49] B. Liao, J. Li, Z. Ju, and G. Ouyang, "Hand gesture recognition with generalized Hough transform and DC-CNN using realsense," in *Proc. 8th Int. Conf. Inf. Sci. Technol. (ICIST)*, Jun. 2018, pp. 84–90.
[50] D.-S. Tran, N.-H. Ho, H.-J. Yang, E.-T. Baek, S.-H. Kim, and G. Lee, "Real-time hand gesture spotting and recognition using RGB-D camera and 3D convolutional neural network," *Appl. Sci.*, vol. 10, no. 2, p. 722, 2020.
[51] R. Azad, M. Asadi-Aghbolaghi, S. Kasaei, and S. Escalera, "Dynamic 3D hand gesture recognition by learning weighted depth motion maps," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 6, pp. 1729–1740, Jun. 2019.
[52] A. Das, I. Tashev, and S. Mohammed, "Ultrasound based gesture recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2017, pp. 406–410.
[53] Y. Zheng, Y. Zhang, K. Qian, G. Zhang, Y. Liu, C. Wu, and Z. Yang, "Zero-effort cross-domain gesture recognition with Wi-Fi," in *Proc. 17th Annu. Int. Conf. Mobile Syst., Appl., Services*, 2019, pp. 313–325.
[54] S. Skaria, A. Al-Hourani, M. Lech, and R. J. Evans, "Hand-gesture recognition using two-antenna Doppler radar with deep convolutional neural networks," *IEEE Sensors J.*, vol. 19, no. 8, pp. 3041–3048, Apr. 2019.
[55] H. R. Lee, J. Park, and Y.-J. Suh, "Improving classification accuracy of hand gesture recognition based on 60 GHz FMCW radar with deep learning domain adaptation," *Electronics*, vol. 9, no. 12, p. 2140, 2020.
[56] M. Chmurski, G. Mauro, A. Santra, M. Zubert, and G. Dagasan, "Highly-optimized radar-based gesture recognition system with depthwise expansion module," *Sensors*, vol. 21, no. 21, p. 7298, 2021.
[57] E. Rahimian, S. Zabihi, A. Asif, D. Farina, S. F. Atashzar, and A. Mohammadi, "FS-HGR: Few-shot learning for hand gesture recognition via electromyography," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 29, pp. 1004–1015, 2021.
[58] Z. Lu, S. Qin, X. Li, L. Li, and D. Zhang, "One-shot learning hand gesture recognition based on modified 3D convolutional neural networks," *Mach. Vis. Appl.*, vol. 30, no. 7, pp. 1157–1180, 2019.
[59] N. Madapana and J. P. Wachs, "Hard zero shot learning for gesture recognition," in *Proc. 24th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2018, pp. 3574–3579.
[60] Z. Fan, H. Zheng, and X. Feng, "A meta-learning-based approach for hand gesture recognition using FMCW radar," in *Proc. Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Oct. 2020, pp. 522–527.
[61] H. Li, W. Dong, X. Mei, C. Ma, F. Huang, and B.-G. Hu, "LGM-Net: Learning to generate matching networks for few-shot learning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 3825–3834.

[62] X. Zeng, C. Wu, and W.-B. Ye, "User-definable dynamic hand gesture recognition based on Doppler radar and few-shot learning," *IEEE Sensors J.*, vol. 21, no. 20, pp. 23224–23233, Oct. 2021.
[63] W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C. F. Wang, and J.-B. Huang, "A closer look at few-shot classification," 2019, *arXiv:1904.04232*.
[64] G. Mauro, M. Chmurski, M. Arsalan, M. Zubert, and V. Issakov, "One-shot meta-learning for radar-based gesture sequences recognition," in *Proc. Int. Conf. Artif. Neural Netw.*, Cham, Switzerland: Springer, 2021, pp. 500–511.
[65] V. C. Chen, *The Micro-Doppler Effect Radar*. Norwood, MA, USA: Artech House, 2019.
[66] R. J. Weber and Y. Huang, "Analysis for Capon and MUSIC DOA estimation algorithms," in *Proc. IEEE Antennas Propag. Soc. Int. Symp.*, Jun. 2009, pp. 1–4.
[67] A. Antoniou, H. Edwards, and A. Storkey, "How to train your MAML," 2018, *arXiv:1810.09502*.
[68] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 1310–1318.
[69] A. Nichol and J. Schulman, "Reptile: A scalable metalearning algorithm," 2018, *arXiv:1803.02999*.

GIANFRANCO MAURO received the B.Sc. degree in electronics engineering and the M.Sc. degree in biomedical engineering—technologies for electronics from the Politecnico di Milano, in 2017 and 2020, respectively. He is currently pursuing the Ph.D. degree with the University of Granada, in collaboration with Infineon Technologies AG. He is working on self-learning for radar-based applications, especially in the fields of activity recognition and health monitoring. His main research interests include few-shot learning, data feature extraction, and the innovative use of radar technologies in the everyday context.

MATEUSZ CHMURSKI received the B.Sc. degree in embedded system design and the M.Sc. degree in artificial intelligence from the Technical University of Łódź (TUL), Łódź, Poland, in 2016 and 2018, respectively, where he is currently pursuing the Ph.D. degree in artificial intelligence on the edge. From 2015 to 2018, he was a Research Assistant with the Laboratory of Infineon Technologies AG, Deggendorf, Germany. Since 2018, he has been employed with the Headquarter of Infineon Technologies AG, Munich, Germany.

LORENZO SERVADEI (Member, IEEE) received the Ph.D. degree *(summa cum laude)* from Johannes Kepler University Linz, in collaboration with Infineon Technologies AG. During his Ph.D. studies, his research interest includes hardware optimization with machine learning. He is currently working as a Senior Staff Machine Learning Engineer at Infineon Technologies AG and lecturing machine learning at the Technical University of Munich. He is an ACM Member.

MANUEL PEGALAJAR-CUELLAR graduated in computer engineering, in 2003. He received the Ph.D. degree, in 2006, with a focus on time series prediction, parameter identification, and neural networks. He is currently a full-time Teacher with the Department of Computer Science and Artificial Intelligence, University of Granada, Granada, Spain. He has worked in multivariate image analysis and real-time control tasks. His research interests include neural and social networks, evolutionary optimization, and fuzzy systems.

DIEGO P. MORALES-SANTOS received the B.Sc., M.Eng., and Ph.D. degrees in electronics engineering from the University of Granada, in 2001 and 2011, respectively. Since 2001, he has been an Associate Professor with the Department of Computer Architecture and Electronics, University of Almeria, before joining the Department of Electronics and Computer Technology, University of Granada, in 2006, where he currently works as a Professor (tenured). He is the Co-Founder of the Biochemistry and Electronics as Sensing Technologies (BEST) Research Group, University of Granada. He has coauthored more than 80 scientific contributions. His current research interests include low-power energy conversion, energy harvesting for wearable sensing systems, and new materials for electronics and sensors.