A Lightweight CNN-Based Solution for Inertial Gesture Recognition on Tiny Edge Devices

A Lightweight CNN-Based Solution for Inertial Gesture Recognition on Tiny Edge Devices

Marco Esposito*, Sara Raggiunto †, Paolo Napoletano‡,
Alberto Belli*, Monica Marconi Sciarroni*, Emanuele Storti*, Paola Pierleoni*
* Department of Information Engineering, Università Politecnica delle Marche, Ancona, Italy
† National Inter-University Consortium for Telecommunications (CNIT),
Research Unit of Università Politecnica delle Marche, Ancona, italy
‡ University of Milan-Bicocca, Milan, Italy
Email: m.esposito@staff.univpm.it

***Abstract*—The use of wearable devices with inertial sensors for gesture recognition is becoming increasingly common in extended reality and remote control applications. Fast and accurate gesture recognition is critical for real-time services such as industrial control and thus drives the shift of computation toward edge computing. This work aims to optimize deep learning models for direct integration into real-world edge devices. In this context, several models based on convolutional neural networks with different configurations are developed and tested through standard performance benchmarks (accuracy and Macro F1 Score). Moreover, in order to decrease models' size with minimal accuracy loss, they were quantized using full-integer quantization. Non-quantized and quantized models have been tested on different boards through the STM32 Edge AI Developer Cloud, as an additional benchmark to verify that the proposed models meet speed and accuracy requirements also on low-cost, low-power edge nodes. The results show that the best lightweight configuration with post-training quantization achieves an inference time of 17.31 ms on the STM32L4R9I-DISCO board, while maintaining an accuracy of 95.95 % ± 0.31 %, competitive against the state of the art and making it suitable for integration into industrial control frameworks.**

***Index Terms*—edge computing, gesture recognition, wearable sensors, Industry 5.0, Inertial Measurement Unit, human-machine interaction**

I. Introduction

Industry 5.0 introduces a paradigm shift to industrial systems by placing humans at the center of increasingly intelligent, connected, and safer environments [1]. Rather than replacing human operators, Industry 5.0 enhances collaboration between people and advanced technologies, such as Artificial Intelligence (AI), industrial machinery, and wearable sensors. Within this framework, gesture recognition based on Inertial Measurement Units (IMUs) emerges as a key enabling technology to enhance human-machine interactions [2]. IMU-based systems, typically worn on the body or integrated into work garments and devices, allow for accurate tracking and interpretation of human movements in real time. This capability is particularly valuable in smart manufacturing settings, where intuitive, hands-free control can significantly improve safety and productivity. By translating natural human gestures into machine-readable commands, gesture recognition systems simplify and enhance communication between workers and industrial systems such as robots and machinery [3]. Both contactless and wearable systems can be employed for gesture recognition [4]. The former includes, for example, RGB camera systems and infrared systems, while examples of the latter are electromyography bands [5] and inertial units. Among wearable systems, IMUs are especially interesting since they are less noisy and suffer fewer placement issues compared to other contact sensing solutions [6]. Different approaches for IMU-based gesture recognition have been proposed over the years. Notable examples are U-WeAr [7] and DeepGesture [8]. The former is a model based on Gated Recurrent Units (GRUs) for joint gesture classification and user recognition, using recurrent layers for gesture recognition, with the learned features also contributing to user classification. A version of the model has also been analyzed for deployment on a smartwatch with WearOS [9]. DeepGesture is a hybrid model using a 2D Convolutional Neural Network (CNN) head with a kernel height of 1 for feature extraction and recurrent GRU layers for feature classification. Other papers propose 1D-CNN model to automatically extract features and gesture recognition from triaxial accelerometer and triaxial gyroscope signals [6], [10], [11]. While there are no standardized gesture dictionaries, U-WeAr has a wide range of gesture classes, encompassing letters, numbers, and common dynamic gestures, for a total of 25 classes, including a rejection class. In contrast to U-WeAr comprehensive dataset, most other notable models are trained and evaluated on a smaller amount of gestures. For example, DeepGesture uses a total of 9 classes. In both cases, reported accuracies above 96% are achieved on their respective datasets.

This work has been partially supported by the PRIN 2022 project "HOMEY: a Human-centric IoE-based Framework for Supporting the Transition Towards Industry 5.0", funded by the European Union - Next Generation EU, Mission 4 Component 1 (code: 2022NX7WKE, CUP: F53D23004340006)

DOI: 10.1109/ETFA65518.2025.11205794

While they provide good performance, the adaptation of these models to edge scenarios requires careful tuning. In fact, edge devices, such as wristbands or other wearable extended reality (xR) equipment, are low-resource and often low-power. Challenges arise when embedding Recurrent-Neural-Network-(RNN)-based models on such devices due to recurrent neural networks' high computational and memory demands [12]. Conversely, 1D-CNNs have proven to be extremely compact in tinyML scenarios, also compared to Long-Short-Term-Memory (LSTM) models [13]. At the same time, they have shown good performance when compared to their recurrent counterparts also in sequence modeling and classification, especially when combined with causality and dilation [14].

In this work, we propose a compact CNN-based architecture for IMU-based gesture recognition, achieving performance comparable to state-of-the-art hybrid models on a unified benchmark dataset (U-WeAr). Our main contribution is demonstrating that causal 1D-CNNs can effectively replace recurrent layers for time series classification while resulting in a lower computational complexity, improved inference latency, and resource-efficient deployment on embedded systems. Various configurations of the convolutional layers are explored, and the robustness of the proposed model is verified through 5-Fold and Leave-One-Subject-Out Cross-Validation techniques. Post-training, full-integer quantization is applied to all models, and their performance is tested on a range of STM32 development boards. While 1D-CNNs have been applied to IMU-based gesture recognition and similar time-series tasks [15], [16], prior work generally focuses on limited gesture vocabularies and pays little attention to edge deployment constraints. In contrast, this work demonstrates that causal 1D-CNNs can accurately classify gestures from large dictionaries while maintaining efficiency suitable for edge scenarios. Furthermore, another added value of this paper was the profiling of the proposed models on real devices. This enables a thorough evaluation and validation of both non-quantized and quantized CNN models on STM32 boards, assessing average inference time, RAM usage, and Flash memory footprint.

II. Methods

This section details the experimental setup, including the proposed 1D-CNN architectures, training methodology, and evaluation metrics used to assess performance on the U-WeAr dataset.

A. 1D-CNN Architecture and Ablation Design

The main motivation of the proposed 1D architecture is to address observed limitations of state-of-the-art models for gesture recognition at the edge, particularly DeepGesture. The main hypothesis is that, by removing the recurrent layers of the original DeepGesture model, which represent a heavy bottleneck for embedded implementation, a convolution-only architecture can be developed, leveraging the feature extraction capabilities of DeepGesture CNN-head while resulting in a lighter model for edge implementation. While DeepGesture's convolutional head utilizes 2D convolutions with a kernel height of 1 (effectively performing 1D temporal convolutions per channel), the proposed model uses native 1D-CNNs. Studies highlight the advantages of 1D-CNNs for time-series data over their 2D counterparts, particularly for low-cost, embedded platforms [17]. The proposed model employs 1D convolutions with causality, which ensures that the model's output at a given timestep depends only on current and past timesteps. In particular, we tested five convolutional variants to evaluate the trade-off between computational efficiency and classification accuracy. All configurations are based on 1D convolutional layers, varying in depth, filter width, and stride. The five configurations are detailed in Table I and described below. Config ID 0 is the deepest (7 convolutional layers) and widest (up to 64 filters) configuration, using alternating strides. Config ID 1 is a shallower (4 convolutional layers) but still wide variant. Config ID 2 is the narrowest, with filter counts from 6 to 32 and 7 convolutional layers. Config ID 3 has the same depth as Config ID 0 but with a slightly narrower filter progression. Finally, Config ID 4, with 4 convolutional layers, adapts the convolutional structure of the original DeepGesture to 1D causal convolutions. All configurations end with a dense layer producing raw logits without softmax activation, as class predictions during inference are made via argmax and the removal of softmax improves quantization stability [18], [19]. Softmax activation is applied during training, which was

TABLE I: Details of the 1D convolutional layer for each configuration of the proposed model under test.

Config ID	Layer #	Filters	Kernel	Stride
0	1	12	5	2
	2	24	3	1
	3	24	3	1
	4	48	3	2
	5	48	3	1
	6	64	3	2
	7	64	3	1
1	1	12	5	2
	2	24	3	1
	3	48	3	2
	4	64	3	1
2	1	6	5	2
	2	12	3	1
	3	12	3	1
	4	24	3	2
	5	24	3	1
	6	32	3	2
	7	32	3	1
3	1	6	5	2
	2	12	3	1
	3	24	3	1
	4	48	3	2
	5	48	3	1
	6	64	3	2
	7	64	3	1
4	1	12	3	2
	2	24	3	2
	3	48	3	2
	4	48	3	1

carried out using Adam optimizer, a batch size of 25, 100 epochs, and categorical cross-entropy loss.

B. Dataset

The U-WeAr dataset has been used for model training and testing. It consists of tri-axial accelerometer and gyroscope data, acquired using a custom inertial measurement unit sensor. A total of 25 classes (24 gestures, 1 rejection class) are recorded from 32 users, with each user performing 5 repetitions of each gesture. A full description of the dataset and acquisition protocol can be found in [7].

C. Evaluation Protocols

The models were evaluated using both 5-Fold Cross-Validation (CV) and Leave-One-Subject-Out (LOSO) CV. For the former, the dataset is randomly split into 80% training and 20% validation, repeating this process across five different folds. Then, the average and standard deviation of performance metrics are computed to assess per-fold average performance and variations across folds. In LOSO CV, each user's data is used once as the validation set while the rest is the training set. Average and standard deviation of metrics is used to evaluate generalization and per-user variation.

D. Metrics

Common metrics are used to evaluate performance in the multiclass classification task (25 classes). In detail, the models are assessed in terms of the accuracy and macro-averaged F1 score.

Accuracy is defined as:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

F1 score for class i is the harmonic mean of precision and recall:

$$\text{F1}_i = 2 \cdot \frac{\text{Precision}_i \cdot \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i}$$

where

$$\text{Precision}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i}, \quad \text{Recall}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FN}_i}$$

Macro-F1 score is then the average over all $C = 25$ classes:

$$\text{Macro-F1} = \frac{1}{C} \sum_{i=1}^{C} \text{F1}_i$$

E. Quantization and Deployment Setup

Post-Training Quantization (PTQ) is performed for each of the proposed model configurations using TFLite converter tool. Namely, Full-Integer (FI) quantization is used with the goal to achieve the highest reduction of model size and inference time. Both outputs and inputs are quantized, using a representative dataset to calibrate the quantization ranges for both. Quantized models are then evaluated using 5-Fold CV.

All models (baseline models and FI quantized models) were benchmarked on the following boards through STM32 Edge AI Developer Cloud: STM32L4R9I-DISCO, NUCLEO-F401RE, STM32F4691-DISCO. In particular, the STM32L4R9I-DISCO board embeds an ultra-low-power STM32L4 series MCU, making it suitable for energy-constrained applications. A recap of each board's main characteristics is in Table II.

III. Results

A. Preliminary Benchmarking

DeepGesture was trained on a limited dictionary of gestures compared to U-WeAR. A preliminary benchmarking was therefore carried out to establish a baseline performance on the U-WeAr dataset before cross-validating the proposed models. Table III shows a performance comparison of different DeepGesture variants, with increasing number of recurrent layers, on a random 80/20 split. The original 4-layer CNN head is maintained across tests (refer to Table I, ConfigID 4). It can be observed how performance does not increase much based on the depth of recurrent layers, but it oscillates between 93% and 95%. U-WeAr shows similar results on user-indepedent validation but with a reduced parameter count of circa 75k parameters [7]. While both hybrid and fully recurrent models achieve competitive accuracy, their overall memory footprint makes them less suitable for deployment on certain resource-constrained edge devices, such as bare metal devices.

B. 1D-CNN Models

Table IV shows 5-Fold CV and LOSO CV results for the different configurations of the 1D CNN models under test. Results clearly show that a purely convolutional architecture can match RNN-based gesture recognition models while also reducing model complexity. All configurations, and particularly Config ID 1, provide a good trade-off between accuracy and size, achieving a LOSO accuracy of more than 94% with a reduced number of parameters compared to the recurrent models. These results support the use of causal 1D-CNNs as an efficient alternative to RNNs for embedded gesture recognition tasks. For comparison, DeepGesture smaller version (3 GRU layers) has more than 133k parameters, while U-WeAr has approximately 75k. All configurations exhibit stable performance in 5-Fold CV with low variance, while LOSO results are slightly more variable due to subject differences, but these variations are in line with previous work using U-WeAr as the benchmark dataset [7]. To assess the main misclassifications, Figure 1 shows the confusion matrices for the 5 configurations, relative to a fold. The slight confusions are consistent, primarily between classes 14 and 23. Those are the '3' and 'hello' gestures from U-WeAr [7], respectively.

C. Quantization and Real-Time Testing

For testing the proposed models for real-time system deployment, those were quantized, cross-validated, and then deployed on a series of reference boards. Table V shows the results for this testing. Across all configurations, FI-PTQ consistently compresses the model by up to 6.7 times in flash (e.g., for Config ID 0 size goes from 159 KB to 74 KB) and reduces RAM usage by over 50%, with

TABLE II: Hardware characteristics of the STM32 boards used for benchmarking.

Board	Core	Frequency	RAM	Flash
STM32L4R9I-DISCO	Arm Cortex-M4	120 MHz	640 KB internal	2048 KB internal, 64 MB external
NUCLEO-F401RE	Arm Cortex-M4	84 MHz	96 KB internal	512 KB internal
STM32F469I-DISCO	Arm Cortex-M4	180 MHz	384 KB internal, 16 MB external	2048 KB internal, 16 MB external

TABLE III: Performance of GRU-based gesture recognition models with varying numbers of recurrent units on a random 80/20 split. Macro-averaged values are shown.

GRUs	Params	Size (MB)	Accuracy (%)	Precision (%)	Recall (%)	Macro-F1 (%)
3	133,361	1.66	94.88	94.73	94.92	94.90
4	158,577	1.97	95.00	95.02	95.09	95.07
5	183,793	2.27	93.13	93.53	93.13	93.21
6	209,009	2.58	95.25	95.36	95.18	95.19
7	234,225	2.89	94.88	94.78	94.95	94.86
8	259,441	3.19	94.25	94.65	94.33	94.35
9	284,657	3.50	94.38	94.41	94.39	94.39

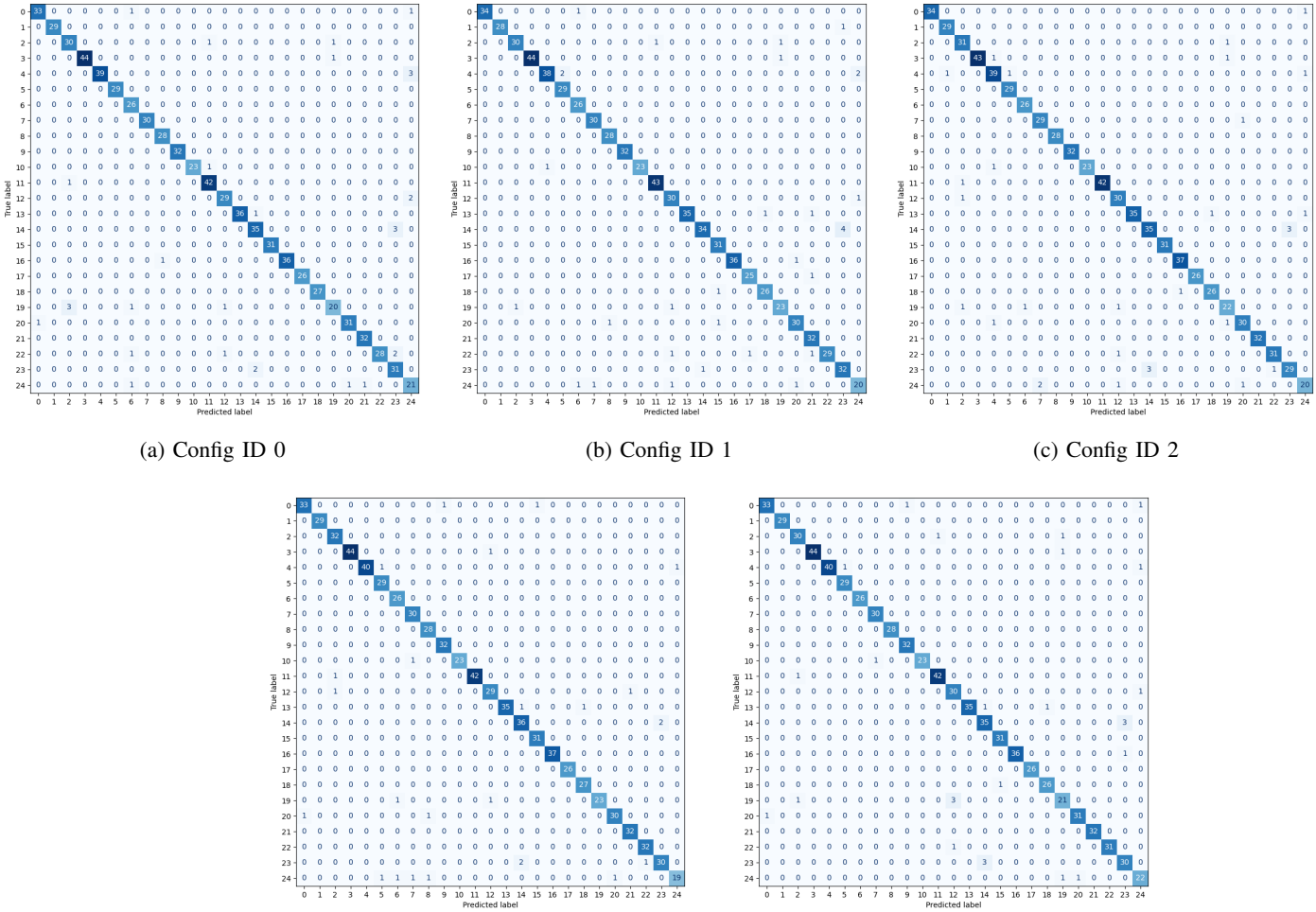
TABLE IV: Accuracy, F1-score, and parameter count for each configuration under LOSO and 5-Fold Cross Validation.

Config ID	Accuracy		Macro-F1		Params
	5-Fold CV	LOSO	5-Fold CV	LOSO	
0	96.50 ($\pm$ 0.57)	94.35 ($\pm$ 6.17)	96.45 ($\pm$ 0.60)	94.04 ($\pm$ 6.53)	37,869
1	96.30 ($\pm$ 0.50)	94.60 ($\pm$ 5.20)	96.24 ($\pm$ 0.57)	94.26 ($\pm$ 5.48)	16,261
2	95.03 ($\pm$ 0.65)	94.43 ($\pm$ 5.00)	94.90 ($\pm$ 0.59)	94.05 ($\pm$ 5.42)	10,331
3	96.45 ($\pm$ 0.60)	94.23 ($\pm$ 5.59)	96.35 ($\pm$ 0.71)	93.88 ($\pm$ 5.89)	36,087
4	96.10 ($\pm$ 0.32)	94.10 ($\pm$ 6.17)	96.01 ($\pm$ 0.40)	93.73 ($\pm$ 6.17)	13,333

TABLE V: 5-Fold Cross-Validation Results (%): Float vs. Quantized Models (mean $\pm$ std)

Config ID	Model	Accuracy (mean$\pm$std)	Macro-F1 (mean$\pm$std)	Size (KB)	RAM (KiB)	Flash (KiB)
0	baseline	96.50 $\pm$ 0.57	96.45 $\pm$ 0.60	585	53.33	159.02
	FI-PTQ	96.58 $\pm$ 0.63	96.53 $\pm$ 0.66	87	26.57	73.71
1	baseline	96.30 $\pm$ 0.50	96.24 $\pm$ 0.57	278	59.28	74.62
	FI-PTQ	96.20 $\pm$ 0.59	96.14 $\pm$ 0.66	33	27.64	47.96
2	baseline	95.03 $\pm$ 0.65	94.90 $\pm$ 0.59	258	29.26	53.65
	FI-PTQ	94.93 $\pm$ 0.77	94.79 $\pm$ 0.72	36	18.96	46.27
3	baseline	96.45 $\pm$ 0.60	96.35 $\pm$ 0.71	564	53.33	152.34
	FI-PTQ	96.43 $\pm$ 0.56	96.35 $\pm$ 0.63	65	28.7	71.91
4	baseline	96.10 $\pm$ 0.32	96.01 $\pm$ 0.40	243	28.17	63.43
	FI-PTQ	95.95 $\pm$ 0.31	95.85 $\pm$ 0.40	30	19.55	45.03

little to no impact on accuracy and F1 score. Config ID 1 achieves a balance between performance and deployability, with only 33 KB model size maintaining 96.2% accuracy, matching the larger configurations. Config ID 4 is the most lightweight, as it requires only 30 KB flash and 19.5 KiB RAM, while still reaching 95.95% accuracy. For comparison, we also applied post-training quantization to the DeepGesture model. Although its accuracy remains competitive (and an average accuracy of $96.6 \pm 0.6\%$.), the resulting quantized model requires approximately 175 KB of flash, significantly larger than any of the proposed models. Due to this increased memory footprint and the lack of clear benefits, we exclude DeepGesture from the subsequent on-board evaluation. Figure 2 shows the confusion matrices for the quantized models for the same fold as Figure 1. Performance remains very well aligned to the original models, and confusions are consistent. All models were also evaluated in terms of inference times on the target boards. Table VI shows the results across the three boards. All proposed models are suitable for real-time deployment in edge-based control scenarios. On the STM32L4R9I-DISCO board, the fastest configuration (Config ID 4) achieves inference times of 17.31 ms. The STM32L4 MCU family is an ultra-low-power series used in embedded platforms for smart industry industrial nodes and wearable devices, such as the



(a) Config ID 0 (b) Config ID 1 (c) Config ID 2

(d) Config ID 3 (e) Config ID 4

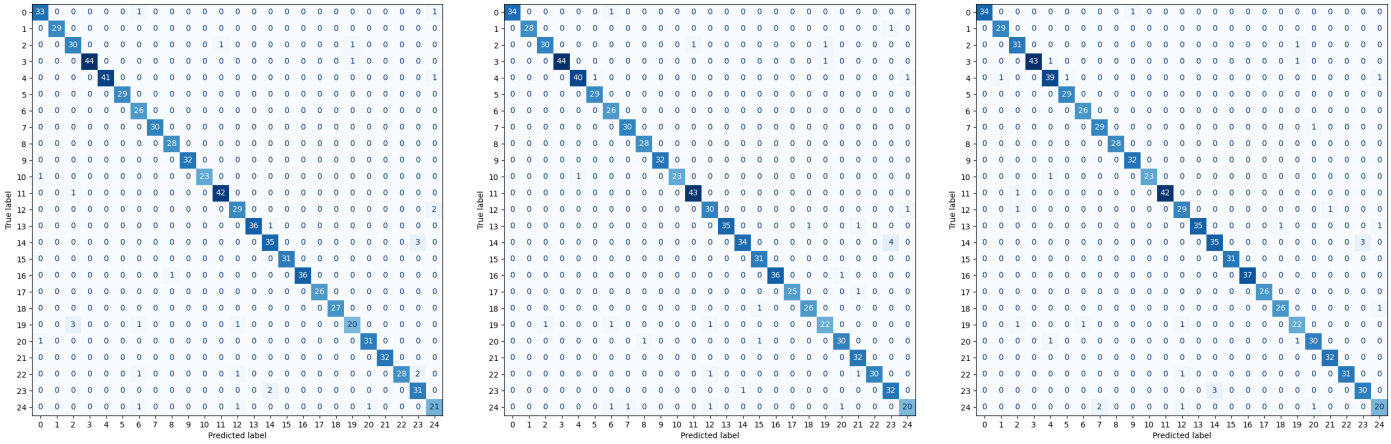
Fig. 1: Confusion matrices for the 5 configurations of 1D-CNN under test.

SensorTile Box by STM32 [20]. Furthermore, these times are well within the constraints of practical control systems, where, for example, usability studies suggest that, for remote device control, execution responsiveness under 200 ms is considered acceptable [21]. For comparison, the 2D equivalent of the smallest model (Config ID 4), which is the same as the original convolutional head of the DeepGesture model[1], requires higher inference time (525.6 ms) on the STM32L4R9I-DISCO board. In summary, our 1D-CNN-based models demonstrate competitive accuracy and F1-scores compared to more complex RNN-based architectures while achieving significantly reduced model size, memory footprint, and very low inference times on edge platforms after quantization, which did not have any significant effect on model accuracy. Config ID 4 achieves the best performance in terms of overall trade-off between memory (30 KiB model size), inference time (17.4 ms), and post-quantization accuracy (95.95 ± 0.31). As a comparison, Config ID 2, which has the lowest parameter count, achieves lower post-quantization accuracy and higher inference time, possibly due to the use of alternating strides, which do not give relevant benefits in terms of accuracy despite their aim to maintain higher temporal resolution of extracted features.

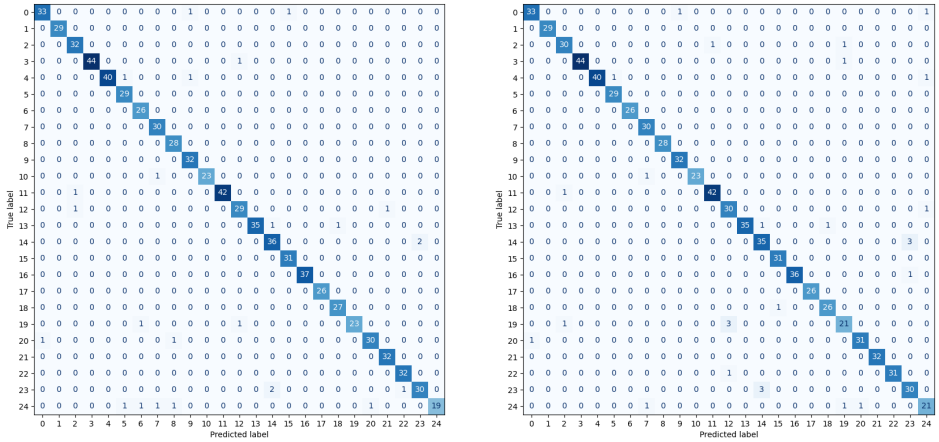
[1]https://github.com/GwangsHong/DeepGesture/tree/master

IV. Conclusion

This study aims to propose a lightweight solution for inertial gesture recognition suitable for edge implementation. Firstly, DeepGesture and its variants were evaluated, showing that the overall memory footprint makes them less suitable for deployment on resource-constrained edge devices. Subsequently, a series of CNN models was developed for the implementation of a gesture recognition system on board low-cost, low-resource wearable devices. The developed models were designed starting from state-of-the-art model and optimizing it for low-latency inference by removing bottlenecks derived from the use of recurrent layer. In particular, five different CNN-based configurations were developed, trained, and tested using the U-WeAr dataset, and were assessed using 5-Fold Cross-Validation and Leave-One-Subject-Out Cross-Validation. Performance was evaluated in terms of accuracy and Macro F1 Score. Results show that a CNN-only model



(a) Config ID 0 (b) Config ID 1 (c) Config ID 2



(d) Config ID 3 (e) Config ID 4

Fig. 2: Confusion matrices for the 5 configurations of 1D-CNN under test after FI-PTQ.

provides high performance for gesture recognition, in both user-dependent and user-independent tasks. Full-Integer, Post-Training Quantizatio was applied to all models, resulting in stable accuracy pre- and post-quantization. This allowed for a comparison between quantized and non-quantized models in terms of size, RAM, and flash memory usage. Benchmarking on STM32 boards proved low inference times across different MCU families, including on the low-power, low-resource STM32L4 family, where the fastest model reaches 17.31 ms of inference time. Future work will involve the creation of a dedicated dataset on STM32 hardware and the integration of user recognition alongside the gesture recognition task that has been tackled in this work. The gesture recognition module will be integrated into a comprehensive Internet-of-Everything (IoE) framework [22] aligned with the principles of Industry 5.0 and designed to facilitate seamless collection, monitoring, and access to data generated by heterogeneous IoT sensors, wearables, and smart objects. In particular, recognized gestures will be transmitted as apposite Bluetooth Low Energy commands and, through a gateway, sent to a dedicated messaging service (e.g., an MQTT broker), which will then dispatch the commands to the appropriate data consumers (e.g., specific devices to control). This architecture will contribute to the creation of human-centric, immersive digital work environments that support advanced human-machine interaction.

REFERENCES

[1] J. Leng, W. Sha, B. Wang, P. Zheng, C. Zhuang, Q. Liu, T. Wuest, D. Mourtzis, and L. Wang, "Industry 5.0: Prospect and retrospect," *Journal of Manufacturing Systems*, vol. 65, pp. 279–295, 2022.

[2] M. Kim, J. Cho, S. Lee, and Y. Jung, "Imu sensor-based hand gesture recognition for human-machine interfaces," *Sensors*, vol. 19, no. 18, p. 3827, 2019.

[3] X. Wang, D. Veeramani, and Z. Zhu, "Wearable sensors-based hand gesture recognition for human–robot collaboration in construction," *IEEE Sensors Journal*, vol. 23, no. 1, pp. 495–505, 2022.

[4] L. Guo, Z. Lu, and L. Yao, "Human-machine interaction sensing technology based on hand gesture recognition: A review," *IEEE Transactions on Human-Machine Systems*, vol. 51, no. 4, pp. 300–309, 2021.

[5] K. Erin, M. Ç. Kutlu, and B. Boru, "Comparison of gesture classification methods with contact and non-contact sensors for human-computer interaction," *Sigma Journal of Engineering and Natural Sciences*, vol. 40, no. 2, pp. 219–226, 2022.

TABLE VI: Evaluation metrics of each configuration on target device

Device	Config ID	PTQ	Inference (ms)
STM32L4R9I-DISCO	0	baseline	392.0
		FI-PTQ	59.87
	1	baseline	233.7
		FI-PTQ	35.70
	2	baseline	133.5
		FI-PTQ	21.09
	3	baseline	340.8
		FI-PTQ	53.49
	4	baseline	107.6
		FI-PTQ	**17.31**
NUCLEO-F401RE	0	baseline	480.0
		FI-PTQ	83.41
	1	baseline	289.1
		FI-PTQ	50.06
	2	baseline	170.4
		FI-PTQ	28.43
	3	baseline	416.8
		FI-PTQ	74.58
	4	baseline	134.4
		FI-PTQ	**23.73**
STM32F469I-DISCO	0	baseline	244.2
		FI-PTQ	40.97
	1	baseline	146.8
		FI-PTQ	24.44
	2	baseline	86.1
		FI-PTQ	14.24
	3	baseline	212.0
		FI-PTQ	36.26
	4	baseline	68.21
		FI-PTQ	**11.64**

[6] A. Dahiya, D. Wadhwa, R. Katti, and L. G. Occhipinti, "Efficient hand gesture recognition using artificial intelligence and imu based wearable device," *IEEE Sensors Letters*, 2024.

[7] S. Bianco, P. Napoletano, A. Raimondi, and M. Rima, "U-wear: User recognition on wearable devices through arm gesture," *IEEE transactions on human-machine systems*, vol. 52, no. 4, pp. 713–724, 2022.

[8] J.-H. Kim, G.-S. Hong, B.-G. Kim, and D. P. Dogra, "deepgesture: Deep learning-based gesture recognition scheme using motion sensors," *Displays*, vol. 55, pp. 38–45, 2018.

[9] A. Colombo, L. Celona, S. Bianco, A. Nocera, and P. Napoletano, "Arm gesture recognition with smartwatches," in *2024 IEEE 8th Forum on Research and Technologies for Society and Industry Innovation (RTSI)*, 2024, pp. 625–629.

[10] Z. He, Q. Li, and Z. He, "Gesture recognition using inertial sensors with 1d convolutional neural network," in *2023 IEEE International Conference on Control, Electronics and Computer Technology (ICCECT)*, 2023, pp. 1456–1460.

[11] S. Mekruksavanich, W. Phaphan, and A. Jitpattanakul, "Sensor-based hand gesture recognition using one-dimensional deep convolutional and residual bidirectional gated recurrent unit neural network," *Lobachevskii Journal of Mathematics*, vol. 46, no. 1, pp. 464–480, 2025.

[12] N. M. Rezk, M. Purnaprajna, T. Nordström, and Z. Ul-Abdin, "Recurrent neural networks: An embedded computing perspective," *IEEE Access*, vol. 8, pp. 57 967–57 996, 2020.

[13] G. Athanasakis, G. Filios, I. Katsidimas, S. Nikoletseas, and S. H. Panagiotou, "Tinyml-based approach for remaining useful life prediction of turbofan engines," in *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2022, pp. 1–8.

[14] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[15] C. Gianoglio, E. Ragusa, R. Zunino, and M. Valle, "1-d convolutional neural networks for touch modalities classification," in *2021 28th IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*, 2021, pp. 1–6.

[16] Z. He, Q. Li, and Z. He, "Gesture recognition using inertial sensors with 1d convolutional neural network," in *2023 IEEE International Conference on Control, Electronics and Computer Technology (ICCECT)*, 2023, pp. 1456–1460.

[17] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, "1d convolutional neural networks and applications: A survey," *Mechanical systems and signal processing*, vol. 151, p. 107398, 2021.

[18] H. Shi, H. Shao, W. Mao, and Z. Wang, "Trio-vit: Post-training quantization and acceleration for softmax-free efficient vision transformer," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 72, no. 3, pp. 1296–1307, 2025.

[19] N. P. Pandey, M. Fournarakis, C. Patel, and M. Nagel, "Softmax bias correction for quantized generative models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 1453–1458.

[20] STMicroelectronics, "STEVAL-STLCS01V1 Data Brief," https://www.st.com/en/evaluation-tools/steval-stlkt01v1.html, [Online].

[21] B. Lin and H. Su, "Experimental evaluation and performance improvement of bluetooth mesh network with broadcast storm," *IEEE Access*, vol. 11, pp. 137 810–137 820, 2023.

[22] M. M. Sciarroni, M. Esposito, P. Pierleoni, and E. Storti, "Monitoring data streams in industry 5.0: a knowledge graph approach," in *2024 IEEE 8th Forum on Research and Technologies for Society and Industry Innovation (RTSI)*. IEEE, 2024, pp. 566–571.