Ternary Neural Networks for Gait Identification in Wearable Devices

2024 IEEE International Workshop on Information Forensics and Security (WIFS) | 979-8-3503-6442-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/WIFS61860.2024.10810715

Giacomo Agnetti
Politecnico di Torino, Italy
giacomo.agnetti@polito.it

Andrea Migliorati
Politecnico di Torino, Italy
andrea.migliorati@polito.it

Daniele Mari
Università di Padova, Italy
daniele.mari@dei.unipd.it

Tiziano Bianchi
Politecnico di Torino, Italy
tiziano.bianchi@polito.it

Simone Milani
Università di Padova, Italy
simone.milani@dei.unipd.it

Enrico Magli
Politecnico di Torino, Italy
enrico.magli@polito.it

***Abstract*—Recently, wearable devices such as smartwatches and smart glasses have considerably risen in popularity. Typically, they are equipped with sensors that can provide a huge flow of potentially useful data for different applications. However, the available computational resources and power supply are rather limited, so complex classification and transmission architectures are not feasible. For these reasons, a growing research interest has been focused on strategies that enhance the deployability of Deep Neural Networks (DNNs) while preserving performance on a given task by reducing the number of network weights and quantizing them. In this paper, we employ Ternary Neural Networks (TNNs), a method combining quantization and parameter pruning. Specifically, we investigate the effectiveness of TNNs for gait biometric identification from wearable sensors, whose application in gait identification systems is largely unexplored. Ternary quantization sets a high number of network parameters falling into an interval to zero, effectively removing them from the network topology, hence enabling great memory and computational efficiency. We provide a lightweight deep classifier model with ternarized weights and activations and train it end-to-end to achieve competitive performance with the state-of-the-art while ensuring remarkably high sparsity rate, at times even greater than 90% (i.e., less than 10% of the parameters remain in the topology). Furthermore, the obtained ternary parameter distribution reaches entropy rates that are significantly lower than 1 bit, allowing further compressibility compared to plain binary neural networks which we also considerably outperform.**

***Index Terms*—Gait identification, Ternary Neural Networks, Embedded Devices, Efficient Deep Learning**

I. Introduction

In recent years, wearable devices such as smartwatches, smart glasses, and wearable sensors have been enhanced by several functionalities thanks to the embedding of an increasing number of sensors and interfaces, including accelerometers, magnetometers, and gyroscopes. Such sensors generate a huge amount of samples that are generally transmitted using Bluetooth Low Energy (BLE). However, the ever-increasing volume of data generated by these devices necessitates efficient strategies for data compression [1] or selective transmission [2] to manage bandwidth and energy consumption effectively, as well as to trigger timely alarms whenever a dangerous situation arises (e.g., an unexpected health issue). Selective transmission, in particular, involves detecting specific events (e.g., heart rate anomalies, falls) to trigger data transmission, allowing more complex algorithms, such as DNNs, to analyze the data and eventually raise an alarm. Even these in-device algorithms for anomaly detection are likely to be based on DNNs due to their modeling capabilities that have led to impressive performance across multiple fields. Nevertheless, edge and wearable devices require architectures able to provide reliable detection performance with limited resources.

To address the challenges of deploying DNNs on resource-constrained devices, two prominent techniques have been developed: network pruning [3] and model quantization [4]. The former entails reducing the number of parameters in the network by removing weights that do not significantly affect performance, while model quantization reduces the precision of the parameters of the network thereby allowing the network to be stored with fewer bits, to run faster, and enjoy lower energy consumption. These methods, paired with specific hardware-aware algorithmic designs, have been widely applied to networks deployed on smartphones and IoT devices. For example, [5] propose an accumulation before multiplication mechanism to reduce the number of multiplications, while Stacker et al. [6] compare ad-hoc libraries for embedded devices, and Yang et al. [7] deploy a Convolutional Neural Network (CNN) on an FPGA for efficient remote sensing. However, these approaches primarily target IoT devices, but fail to adequately address personal wearable devices as the computational requirements need to be utterly reduced (ultra-lightweight and low-power equipment to prevent user discomfort). For example, Khan et al. [8] use a Raspberry Pi to run their algorithm for facial recognition using smart glasses. However, although this requires data transmission, it is not a problem in the considered scenario since transmission is only required whenever a photo is taken by the glasses which happens sporadically. Further on, many biometric and health-related apps continuously acquire data such as heart rate, Inertial Measurement Unit (IMU) samples, and oxygenation levels from sensing devices, and they stream them to a central processing unit (i.e., the smartphone or the connected platform). Such solutions design their classification networks for highly-performing smartphones [9]–[12], without focusing on specific low-power sensing device and computational requirements are satisfied only by keeping the number of layers and parameters low, which is generally less effective than pruning and quantizing weights.

In this paper, network quantization is applied to a gait identification model to enhance its deployability and reduce both size and energy requirements. Following the recent trends in model quantization, TNNs [13]–[17] are employed. In these models weights and activations are quantized on three levels $(+1, 0, -1)$ as opposed to standard Binary Neural Networks (BNNs) where only two levels are considered. The advantage of TNNs is that parameters that fall in the 0 interval are removed from the topology, promoting the sparsity of the deep model. Moreover, other operations can be implemented by simple binary operators that are generally less computationally intensive than multiplications which are commonly performed in matrix multiplication. Additionally, using ternarized parameters makes the model compatible with dedicated deep-learning hardware [13]. For this reason, TNNs are particularly relevant for wearable and portable devices due to their limited computing performance and storage space, as well as to the need for low energy consumption to avoid discomfort due to battery limitations. Moreover, low-dimensionality networks can be easily shared, updated, and transmitted to the network: this enables sharing network templates in place of acquired sample series (in a Federated Learning way) so that user's privacy can be preserved. While TNNs have been applied to image classification problems and large language models, their effectiveness on biometric signals still remains largely unexplored. To the best of our knowledge, this work is one of the first to explore the impact of network compression techniques on biometric signal processing, utilizing ternary neural networks to achieve significant energy efficiency.

The main contributions of this work can be summarized as follows: *(i)* we introduce a lightweight deep classifier for biometric data with ternarized weights and activations, showing the effectiveness of TNNs for classification also in the field of gait recognition and identification; *(ii)* our design enjoys a remarkable reduction in resource consumption due to our ternary design which jointly allows for very high sparsity rates and minor decrease in classification accuracy when compared to the full-precision (FP) baseline; *(iii)* our TNNs simultaneously outperform its binary counterpart and reach entropy rates that are significantly lower than 1 bit, offering great advantages in further compressibility compared to a plain BNN; *(iv)* we test our method on a challenging biometric dataset and show its competitive performance compared with state-of-the-art techniques.

II. Background

This section reviews the most relevant works related to gait identification and authentication and model quantization.

A. Gait Identification

Data acquired from wearable devices has often been used to tackle biometric problems such as gait identification and authentication, and activity classification [18]. Gait identification is the task of recognizing an individual from a given group of users based on its gait, which can be seen as a closed set classification problem. On the other hand, gait authentication requires detecting if the considered gait signal belongs to a given person, which can be seen as a one-vs-all classification scenario. Generally, this was done by applying signal-processing techniques and template matching. In this work, the focus is on IMU-based gait identification schemes, i.e., sensors acquire samples from accelerometer and gyroscope in a wearable device (e.g., smartphone, smartwatch, pin-based sensors) and an algorithm uses that data to solve the task.

In [19], authors use the random projections method to perform dimensionality reduction on the data and thus obtain a distribution of the gait features. This distribution is then compared against a known user distribution, and the authentication is successful if the Jaccard distance between the two is below a certain threshold. More recently, DNNs based approaches have risen in popularity. For example, in [9], the authors extract orientation-independent walking cycles and then use a CNN for feature extraction and an SVM after the last layer for classification. Subsequently, in [10] the authors use a combination of CNNs and RNNs to address user identification and authentication, showing improved results compared to competitors. Adel et al. [20] exploit siamese neural networks for user authentication, while Al et al. [21] analyze the performance of LSTMs for user identification, demonstrating the strong modeling power of this architecture for this task. Other works use multiple feature extractors to capture diverse information from the data. One example is Rifaat et al. [22] where the features produced by an LSTM and a Fully Connected (FC) network are merged to obtain a more discriminative representation that can then be used for classification, similarly [23] exploits a multi-feature extractor based network.

B. Ternary Neural Networks

When it comes to quantizing DNNs, the most used quantization function is $Q(r) = \text{Int}(r/s) - z$, where r is the original parameter, $\text{Int}(\cdot)$ is the rounding function, s is a scaling factor, and z represents the origin point. However, the quantization function introduces non-differentiability in the derivative chain during the back-propagation step and gradients computation. To overcome this challenge, [24] introduced the *Straight Through Estimator* (STE) that replaces the non-differentiable quantization function with the identity function during backpropagation, effectively bypassing it.

BinaryConnect [25] was a groundbreaking effort in network compression, introducing a novel architecture that utilized 1-bit parameters. This set the stage for numerous studies exploring low bit-width quantization, including 1-bit binary

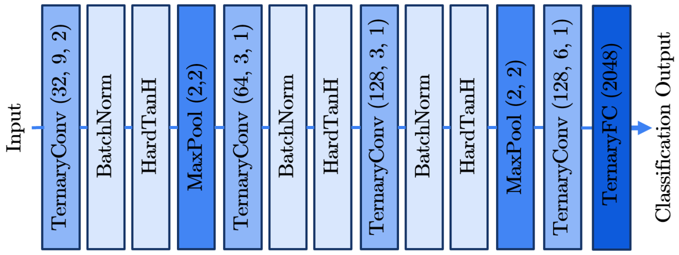

Fig. 1. Diagram of the employed deep architecture. For TernaryConv layers, the number of channels, the kernel size, and the stride size are reported. For MaxPool layers, the kernel and stride size are indicated. For TernaryFC, the number of hidden units is reported.

and 2-bit ternary parameters such as [13] where ternary networks were introduced. Here, weights are quantized through a thresholding function, with the positive threshold parameter calculated by minimizing the Euclidean distance between the quantized parameter and the full-precision version. Trained Ternary Quantization (TTQ) was proposed in [14], featuring layer-dependent quantization values. Furthermore, [26] presented a ternary approach aiming to sparsify and quantize the architecture with a two-stage process composed of compound model scaling, which involves constructing a super-network from a pre-trained model by scaling its dimensions to create an over-parameterized architecture, and quantization, where ternary values are assigned based on a novel cost function featuring an entropy term to promote sparsity in the architecture. FATNN [15] leveraged the ternary representation through a series of bitwise operations, achieving a 2× reduction in computational complexity compared to plain TNNs. Finally, TRQ [16] redefined thresholding operations by generating ternary weights as a combination of binarized stem and residual components, accomplished by recursively quantizing full-precision parameters.

III. Proposed Method

This section presents the proposed method and the reasoning behind the adopted ternarization scheme. First, the developed technique is explained and then the optimal design parameters are found. Finally, the effect of ternarization on sparsity is investigated, and finally, a design choice that ensures great sparsity rates while preserving performance is devised.

A. Ternarization

As anticipated, the ternary values $\{-1, 0, +1\}$ are employed. The parameters and activations are quantized via the following thresholding operation:

$$\theta_q = Ternarize(\theta) = \begin{cases} +1 & \text{if } \theta > \Delta \\ 0 & \text{if } |\theta| \leq \Delta \\ -1 & \text{if } \theta < -\Delta, \end{cases} \tag{1}$$

where θ indicates the parameter and θ_q its quantized version. In this work, we employ STE to train the considered models. Specifically, an STE version that accounts for the gradient saturation effect is used as proposed by [27], such that $g_r = g_q \times \mathbb{1}_{|r| \leq 1}$, where g_r is the gradient of the loss with respect to r, and g_q is an estimator of the gradient $\frac{\partial L}{\partial q}$. In more detail, q is a non-smooth clipping defined as $q = \mathrm{Ternarize}(clip(r, -1, 1))$, where $\mathrm{Ternarize}(r)$ is as in Eq. 1. This forces the parameters to stay confined within the $[-1, +1]$ interval before quantization. The proposed ternarization scheme differs from a static binary quantization framework that quantizes to $-1, +1$ [25], [27] as it allows parameters previously removed from the topology to be reintroduced into the network if this results in reduced training loss. In other words, weight updates determine which parameters should be set to zero (i.e., which nodes should be pruned) to optimize classification accuracy and sparsity.

B. Δ Growth Regimes

The initial experimental results suggested increasing the threshold Δ during training to achieve higher sparsity rates rather than using constant Δ values. For this reason, Δ growth regimes were introduced as follows:

$$\Delta_n = min\left(\Delta_0 + (\Delta_{max} - \Delta_0) f\left(\mathrm{n}\right), \Delta_{max}\right), \tag{2}$$

which represents how Δ changes as a function of the training epoch ($epoch$). In more detail, Δ_0 and Δ_{max} respectively indicate the minimum and maximum values for Δ, n is the current epoch and N is the epoch number when Δ will reach its maximum value, finally, $f(n)$ is a function of the training epoch. Specifically, different functions $f(n)$ were tested, i.e., the constant ($f(n) = 1$) the linear ($f(n) = n/N$), square ($f(n) = (n/N)^2$), square root ($f(n) = \sqrt{(n/N)}$), and logarithmic ($f(n) = \log(n+1)/log(N+1)$) functions, as shown in Fig. 2. Also, Δ_{max} is crucial in determining the critical working point beyond which too many parameters are set to 0 and performance collapses. Indeed, a higher Δ value leads to a greater number of parameters being zeroed out. However, if Δ becomes too large, it can cause excessively high sparsity rates, which may result in fluctuating performance or even a substantial drop in classification accuracy. Therefore, a correct selection of the function f is crucial. Specifically, the reason for introducing threshold growth regimes is that typically, during the initial training iterations, a large portion of the parameters is already quantized to either -1 or $+1$, as opposed to their full-precision counterparts which are allowed to fluctuate inside the $[-\Delta, \Delta]$ interval. Therefore, implementing a Δ threshold growth regime enables the interval to accommodate the rapid changes in parameter values, ensuring the ternarization threshold adapts and facilitates model sparsification throughout the training process.

C. Network Structure

Fig. 1 shows the deep model employed to tackle gait signal classification. A CNN composed of three convolutional layers and a FC layer performing classification on the last convolutional feature maps. However, to adjust with the considered ternarization scheme, we rely on *TernaryConv* layers in place of plain convolutional layers, and on a *TernaryFC* layer instead of FC one. *TernaryConv* and *TernaryFC* are by all measures

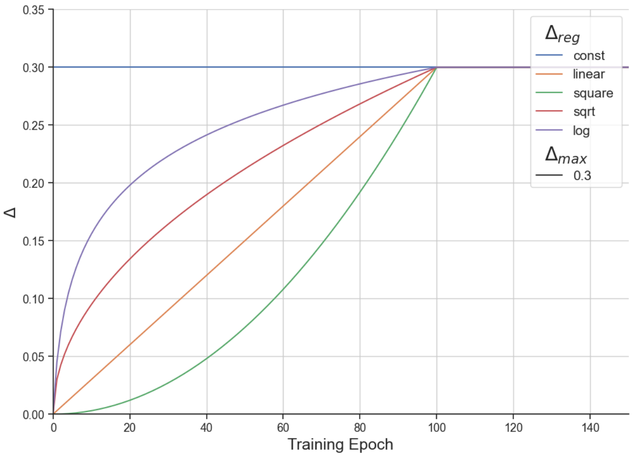

Fig. 2. Examples of Δ growth regimes for different f functions. In this plot, the growth regimes for $\Delta_{max} = 0.3$, reached at the 100^{th} training epoch are shown.

TABLE I

Best test accuracy, sparsity rate, and entropy comparison for different Δ growth regimes on the whuGAIT dataset for the considered network structure. In the top half, we show results for competing state-of-the-art techniques, obtained from different FP baselines. In the bottom half of the table, we report the full-precision (FP) and Binary baselines for our method. Note that the number of parameters for ternary models is obtained by multiplying the total number of parameters by the sparsity rate.

	Acc.	**Sparsity**	**Entropy**	**Acc. Diff.**	**Params.**
	(%)	(%)	(bits/sym)	(%)	(#)
CNN+LSTM [10]	93.52	-	-	-	-
IdNet [9]	92.91	-	-	-	-
DeepConvLSTM [29]	92.25	-	-	-	$>$ 996K
MFEBP [23]	95.38	-	-	-	-
FCN-BiLSTM [22]	95.27	-	-	-	2.89M
BNN Baseline [25]	92.41	-	1	-1.86	372K
Ours (FP baseline)	94.27	-	-	-	372K
Ours (TNN, Δ *const*)	92.39	91.2	0.54	-1.88	**33K**
Ours (TNN, Δ *linear*)	92.02	87.2	0.65	-2.25	48K
Ours (TNN, Δ *square*)	92.78	**91.3**	**0.53**	-1.49	32K
Ours (TNN, Δ *sqrt*)	92.68	85.2	0.75	-1.59	55K
Ours (TNN, Δ *log*)	**93.01**	91.1	0.55	**-1.26**	34K

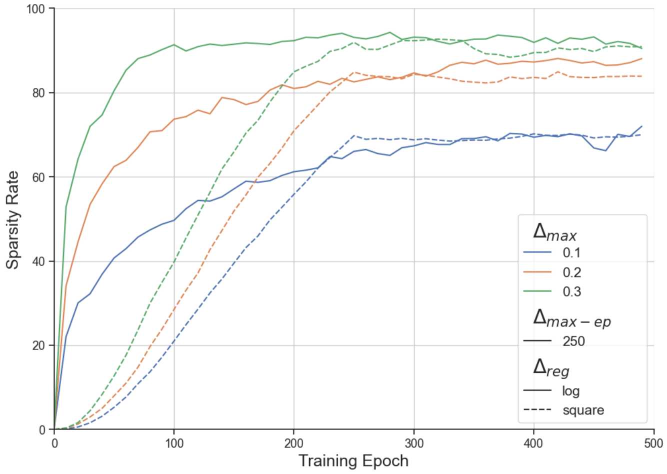

Fig. 3. Examples of how the sparsity rate changes with the training epoch for two different growth regimes (*square* and *log*) and three different Δ_{max} values ($\Delta_{max} = 0.1, 0.2, 0.3$), reached at the 250^{th} training epoch. For $\Delta_{max} > 0.3$ the training becomes unstable and leads to sub-optimal results, as shown in Fig. 4

identical to classic layers, but they perform ternarization of weights and activations on the forward pass, and STE on the backward pass. As explained in Sec. II-B, STE allows to bypass the non-differentiability of the ternarization function (Eq. 1) and to train the deep model. Every *TernaryConv* layer is followed by a *BatchNorm* layer [28] and a *HardTanH* non-linearity, which is preferred over a classic *ReLU* when dealing with quantized parameters [27]. Max-pooling is also introduced to reduce the spatial resolution of the features. In the following, results obtained also on the full-precision (*W32A32*) and binary (*W1A1*) baselines are shown. The former refers to a deep model where no ternarization is performed, therefore every layer is a plain convolutional or fully-connected one. The latter instead indicates that binarization of the weights and activations via *BinaryConv* and *BinaryFC* layers are employed. These are identical to *TernaryConv, TernaryFC* except the ternarization function is replaced by 1-bit quantization [25].

IV. Experiments

In this section, the training and testing setups and the evaluation metrics are described. The proposed method achieves competitive classification results compared with the FP baseline while ensuring a significantly lower resource consumption.

A. Dataset and Setup

Although the proposed ternarization framework can be potentially applied to any biometric signal of choice, it is evaluated on IMU signals from smartphones in the context of gait identification. Gait identification offers the advantage of being unobtrusive. Also, its dynamics can be inexpensively collected via inertial sensors such as gyroscopes or accelerometers which are typically integrated into portable and wearable devices (e.g., smartphones, smartwatches, body sensors). The models are trained and tested on the *whuGAIT* dataset [10] which studies gait recognition using smartphones in the wild by collecting inertial gait data under unconstrained conditions without knowing more about the user's activities. Specifically, its subset called *Dataset 1* is considered, composed of 36844 gait samples divided among 118 different users, and split into 33104 training and 3740 test samples without overlap between the two subsets. Specifically, for each subject, the first 90% samples are used for training while the remaining 10% for test. The proposed lightweight model is composed of around 372000 parameters, as outlined in Fig. 1. The models are trained in batches of a fixed size of 200, with a learning rate set to 0.01. We employ the Adam [30] optimization algorithm to ensure better convergence. The network parameters are initialized by sampling from a normal distribution $\theta \sim \mathcal{N}\left(0, \sqrt{\frac{2}{n}}\right)$, where n is the number of parameters in the convolutional

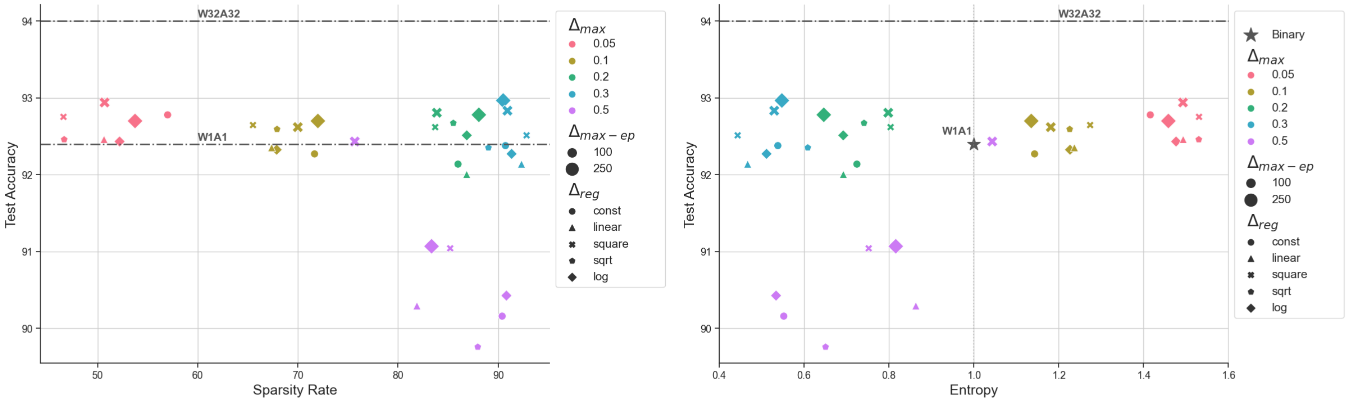

Fig. 4. Test accuracy against sparsity rates (*left*) and test accuracy against entropy (*right*). The charts collect performance on the whuGAIT dataset [10] for different Δ_{max} (reached either at the 100^{th} or 250^{th} training epoch, $\Delta_{max} - ep$) and Δ growth regimes. The graphs also indicate the full-precision (W32A32) and binary (W1A1) baselines (dotted lines) on the same dataset. In the left plot, the best performance is located in the top-right corner, corresponding to high classification accuracy and high sparsity rates. The right plot is essentially a mirrored version of the left one, hence the best models are situated in the top-left area.

kernel. All models are implemented in PyTorch [31] and run on NVIDIA GeForce GTX Titan X GPUs.

B. Evaluation Metrics

The proposed method is evaluated using three metrics: *Top-1 classification accuracy:* the percentage of correctly predicted users on the previously unseen test set; *Sparsity:* the percentage of weights that are removed from the topology of the model; *Entropy:* the average number of bits necessary to represent the information content of the distributions of the ternarized weights, expressed in bits/symbol.

C. Results

Table I reports the best classification accuracy, sparsity rates, and entropy values obtained with the proposed method for different Δ growth regimes. In the top half, we show results for state-of-the-art techniques, obtained from different FP baselines. In the bottom half we also include the FP and Binary baselines for the proposed method. The model performance is comparable with competing methods despite its lightweight design ($372k$ parameters for the FP baseline), while at the same time ensuring sparsity rates even greater than 90%. For example, assuming a sparsity rate equal to 90%, only around $37k$ parameters would remain in the network topology, yielding great model compression. Looking at Table I, one can observe that keeping the Δ interval amplitude constant during training can lead to great sparsity rates, but yields sub-optimal performance in terms of classification accuracy compared to other Δ growth regimes, justifying our design choice to increase the Δ interval size as the training converges.

Figure 3 reports an example of how the sparsity rates change with the training epoch for two different growth regimes (*square* and *log*) and three different Δ_{max} values ($\Delta_{max} = 0.1, 0.2, 0.3$), reached at the 250^{th} training epoch. When $\Delta_{max} > 0.3$, the training becomes unstable and leads to sub-optimal results. For the sake of clarity, $\Delta_{max} > 0.3$ results have not been included in Fig. 3. However, they appear in Fig. 4 where multiple runs are presented to provide a wide-range evaluation of the proposed method. Fig. 3 only shows the *square* and *log* Δ growth regimes as they yield the best results overall both for classification accuracy and sparsity rates. Indeed, by looking at Fig. 3, it is interesting to observe how the *log* growth regime enables greater sparsity rates than the *square* regime early in the training while reaching classification accuracy towards the end.

Finally, Figure 4 shows classification accuracy on the whuGAIT dataset [10] test set against the sparsity rate (*left*) and against the corresponding entropy (*right*), for different Δ_{max} (reached either at the 100^{th} or 250^{th} training epoch $\Delta_{max} - ep$) and Δ growth regimes. Fig. 4 also includes the full-precision (W32A32) and binary (W1A1) baselines (dotted lines). With classification accuracy reaching 93% (less than 1.5% reduction compared with the FP baseline) and sparsity exceeding 91%, the logarithmic growth simulations produce the best results among the considered growth regimes. Also, as anticipated, the best results are obtained consistently for $\Delta_{max} = 0.3$. The highest sparsity rates yield the highest classification accuracy (although by a small margin) thanks to the fact that sparsification has a strong regularizing effect which reduces overfitting, i.e. better performance on the test set. However, high sparsity rates may also lead to greater variability in performance due to the greatly reduced number of parameters. Further, it is worth noticing that the majority of the ternarized models we obtain significantly outperform their binary counterparts in terms of classification accuracy, while also yielding entropy rates lower than 1, showing that there is a clear advantage in employing our ternary framework instead of plain binary networks.

V. Conclusions

In this paper, a ternarization scheme is proposed and applied to a deep neural network to classify gait signals. The framework allows dimensionality and energy requirements reduction since a significant amount of parameters are removed from the network topology. This promotes weight sparsity and provides clear advantages in terms of computational complexity.

The lightweight design shows competitive performance with respect to the full-precision deep model as well as competing gait classification techniques while reaching sparsity rates greater than 90%. This feature translates to entropy rates significantly lower than 1, enabling further compressibility of the network. The method can potentially be applied to the classification of more biometric signals, both in the closed-set and open-set scenarios. These investigations are left as future work, as well as investigating classification performance on other gait biometric datasets.

Acknowledgments

This work was partially supported by the European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, with a partnership on "Telecommunications of the Future" (PE00000001 - program "RESTART"). This study was also carried out within the Future Artificial Intelligence Research (FAIR) and received funding from the European Union Next-GenerationEU (PNRR - Piano Nazionale di Ripresa e Resilienza - Missione 4, Componente 2, Investimento 1.3 - D.D. 1555 11/10/2022, PE00000013). This manuscript reflects only the authors' views and opinions, neither the EU nor the European Commission can be considered responsible for them. Daniele Mari's activities were supported by Fondazione CaRiPaRo under the grants "Dottorati di Ricerca" 2021/2022.

References

[1] U. Jayasankar, V. Thirumal, and D. Ponnurangam, "A survey on data compression techniques: From the perspective of data quality, coding schemes, data type and applications," *Journal of King Saud University-Computer and Information Sciences*, vol. 33, no. 2, pp. 119–140, 2021.

[2] R. Arroyo-Valles, A. G. Marques, and J. Cid-Sueiro, "Optimal selective transmission under energy constraints in sensor networks," *IEEE Transactions on Mobile Computing*, vol. 8, no. 11, pp. 1524–1538, 2009.

[3] H. Cheng, M. Zhang, and J. Q. Shi, "A survey on deep neural network pruning-taxonomy, comparison, analysis, and recommendations," *arXiv preprint arXiv:2308.06767*, 2023.

[4] B. Rokh, A. Azarpeyvand, and A. Khanteymoori, "A comprehensive survey on model quantization for deep neural networks in image classification," *ACM Transactions on Intelligent Systems and Technology*, vol. 14, no. 6, pp. 1–50, 2023.

[5] X. Li, X. Gong, D. Wang, J. Zhang, T. Baker, J. Zhou, and T. Lu, "Abm-spconv-simd: Accelerating convolutional neural network inference for industrial iot applications on edge devices," *IEEE Transactions on Network Science and Engineering*, 2022.

[6] L. e. a. Stäcker, "Deployment of deep neural networks for object detection on edge ai devices with runtime optimization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 1015–1022.

[7] R. Yang, Z. Chen, B. Wang, Y. Guo, and L. Hu, "A lightweight detection method for remote sensing images and its energy-efficient accelerator on edge devices," *Sensors*, vol. 23, no. 14, p. 6497, 2023.

[8] S. Khan, M. H. Javed, E. Ahmed, S. A. Shah, and S. U. Ali, "Facial recognition using convolutional neural networks and implementation on smart glasses," in *2019 international conference on information science and communication technology (ICISCT)*. IEEE, 2019, pp. 1–6.

[9] M. Gadaleta and M. Rossi, "Idnet: Smartphone-based gait recognition with convolutional neural networks," *Pattern Recognition*, vol. 74, pp. 25–37, 2018.

[10] Q. Zou, Y. Wang, Q. Wang, Y. Zhao, and Q. Li, "Deep learning-based gait recognition using smartphones in the wild," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3197–3212, 2020.

[11] I. Koo, Y. Park, M. Jeong, and C. Kim, "Contrastive accelerometer–gyroscope embedding model for human activity recognition," *IEEE Sensors Journal*, vol. 23, no. 1, pp. 506–513, 2022.

[12] E. Hysenllari, J. Ottenbacher, and D. McLennan, "Validation of human activity recognition using a convolutional neural network on accelerometer and gyroscope data," *German Journal of Exercise and Sport Research*, vol. 52, no. 2, pp. 248–252, 2022.

[13] B. Liu, F. Li, X. Wang, B. Zhang, and J. Yan, "Ternary weight networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5.

[14] C. Zhu, S. Han, H. Mao, and W. J. Dally, "Trained ternary quantization," in *International Conference on Learning Representations*, 2017. [Online]. Available: https://openreview.net/forum?id=S1_pAu9xl

[15] P. Chen, B. Zhuang, and C. Shen, "Fatnn: Fast and accurate ternary neural networks," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 5199–5208. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/ICCV48922.2021.00517

[16] Y. Li, W. Ding, C. Liu, B. Zhang, and G. Guo, "Trq: Ternary neural networks with residual quantization," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 10, pp. 8538–8546, 2021. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/17036

[17] S. Ma, H. Wang, L. Ma, L. Wang, W. Wang, S. Huang, L. Dong, R. Wang, J. Xue, and F. Wei, "The era of 1-bit llms: All large language models are in 1.58 bits," *arXiv preprint arXiv:2402.17764*, 2024.

[18] W. Sousa Lima, E. Souto, K. El-Khatib, R. Jalali, and J. Gama, "Human activity recognition using inertial sensors in a smartphone: An overview," *Sensors*, vol. 19, no. 14, p. 3213, 2019.

[19] R. Damaševičius, R. Maskeliūnas, A. Venčkauskas, and M. Woźniak, "Smartphone user identity verification using gait characteristics," *Symmetry*, vol. 8, no. 10, p. 100, 2016.

[20] O. Adel, M. Soliman, and W. Gomaa, "Inertial gait-based person authentication using siamese networks," in *2021 International joint conference on neural networks (IJCNN)*. IEEE, 2021, pp. 1–7.

[21] E. Al-Mahadeen, M. Alghamdi, A. S. Tarawneh, M. A. Alrowaily, M. Alrashidi, I. S. Alkhazi, A. Mbaidin, A. A. Alkasasbeh, M. A. Abbadi, and A. B. Hassanat, "Smartphone user identification/authentication using accelerometer and gyroscope data," *Sustainability*, vol. 15, no. 13, p. 10456, 2023.

[22] N. Rifaat, U. K. Ghosh, and A. Sayeed, "Accurate gait recognition with inertial sensors using a new fcn-bilstm architecture," *Computers and Electrical Engineering*, vol. 104, p. 108428, 2022.

[23] S. Shen, S.-S. Sun, W.-J. Li, R.-C. Wang, P. Sun, S. Wang, and X.-Y. Geng, "A classifier based on multiple feature extraction blocks for gait authentication using smartphone sensors," *Computers and Electrical Engineering*, vol. 108, p. 108663, 2023.

[24] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," *arXiv preprint arXiv:1308.3432*, 2013.

[25] M. Courbariaux, Y. Bengio, and J. David, "Binaryconnect: Training deep neural networks with binary weights during propagations," *Advances in neural information processing systems*, vol. 28, 2015.

[26] A. Marban, D. Becking, S. Wiedemann, and W. Samek, "Learning sparse ternary neural networks with entropy-constrained trained ternarization (ec2t)," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020, pp. 3105–3113. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/CVPRW50498.2020.00369

[27] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks: Training deep neural networks with weights and activations constrained to + 1 or -1," *arXiv preprint arXiv:1602.02830*, 2016.

[28] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. pmlr, 2015, pp. 448–456.

[29] F. J. Ordóñez and D. Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, p. 115, 2016.

[30] P. K. Diederik, "Adam: A method for stochastic optimization," 2014.

[31] A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, 2019, pp. 8024–8035.