

From Cloud-Heavy to Edge-Ready: Self-supervised Transfer-efficient Emotion Recognition

Junjiao Sun*, José Miranda*, Jorge Portilla*, Andrés Otero*

Email: *junjiao.s@upm.es, *jose.miranda@upm.es, *jorge.portilla@upm.es, *joseandres.otero@upm.es

*Center of Industrial Electronics (CEI), UPM, Spain

Abstract—Deploying AI-based emotion recognition at the edge enables real-world applications but is constrained by data scarcity, heavy models, hardware limits, and privacy issues. To overcome these, we propose CHEER (Cloud-HEavy to Edge-Ready), a self-supervised, transfer-efficient framework where the cloud pre-trains lightweight graph-based encoders using unlabeled data, stores them as frozen models, and deploys only the needed encoder. New users are locally matched via centroids of clusters, and a small on-device classifier is trained with minimal labeled data, reducing computation, memory, and energy use while preserving privacy. Experimental results in the WEMAC and WESAD datasets show an accuracy of 78.19% and 80.08% at the edge on a NVIDIA Jetson Orin Nano. Moreover, CHEER achieves more than 60% reduction in model size, and lowers both peak RAM usage and energy consumption by more than 50% compared to the state-of-the-art.

Index Terms—Edge AI, Transfer-efficient deployment, Self-supervised Learning, Graph neural network

I. INTRODUCTION

THE growing integration of AI across cloud, fog, and edge infrastructures is reshaping data processing in distributed systems [1]. Computation is increasingly shifted from centralized servers to the edge, enabling lower latency, reduced bandwidth usage, and enhanced privacy. However, despite growing global efforts toward trustworthy cloud–edge infrastructures, deploying learning models efficiently across this continuum still faces significant methodological challenges. Existing approaches face three main challenges: (1) reliance on large labeled datasets for cloud training [2] [3], (2) redundant full-model transfers to the edge for each new user, causing high communication and energy costs [4], and (3) edge-to-cloud data exchange that compromises privacy [5] [6]. These bottlenecks persist across domains and remain unresolved in prior works such as CLEAR [7].

On this basis, we propose CHEER (Cloud-HEavy to Edge-Ready), a self-supervised, transfer-efficient framework that overcomes these limitations. CHEER pre-trains lightweight Graph Convolutional Network (GCN) Deep Learning (DL) encoders in the cloud using unlabeled structured features, transfers only compact cluster centroids for local assignment, and trains a few-shot classifier at the edge with minimal labeled data, eliminating cloud dependence and preserving user privacy.

The original contributions of this paper are the following.

- A self-supervised and transfer-efficient framework for cloud–edge learning that addresses labeling, transfer, and privacy bottlenecks identified in prior works.

- Integration of a centroid-based model selection mechanism that is executed at the edge and enables communication and energy savings by postponing model downloads until after on-device cluster assignment.

II. PROPOSED METHODOLOGY

CHEER is designed for cloud–edge scenarios and unlabeled multivariate signals. In the cloud, cluster-specific GCN encoders are self-supervised and stored with centroid metadata. During deployment, a new user is locally matched to a cluster via centroids, the corresponding frozen encoder is transferred, and a few-shot classifier is trained on-device. The CHEER pipeline is shown in Fig. 1.

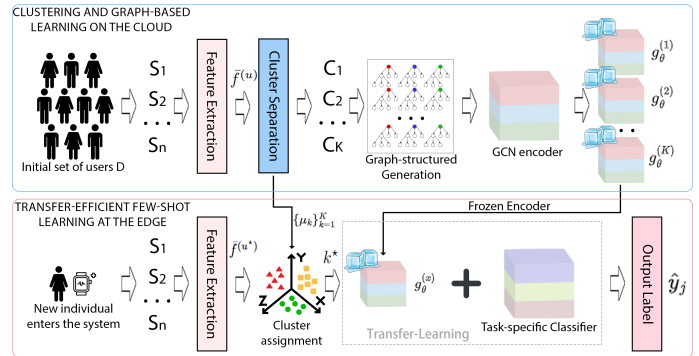


Fig. 1. Overview of the CHEER system: in the cloud, features are extracted, clustered, and used for self-supervised GCN training; at the edge, new users receive the corresponding frozen encoder and train a local classifier.

Cloud pre-training: Let us assume an unlabeled dataset ($\mathcal{D} = \{x^{(u)}\}_{u=1}^N$, $x^{(u)} \in \mathbb{R}^{T \times S}$), for a total N volunteers, where $x^{(u)}$ denotes the multivariate time series of user u , with T time samples and S modalities (the different types of data sources or feature channels). From each signal, a feature vector $\bar{f}^{(u)} \in \mathbb{R}^F$ is extracted, comprising a total of F features (e.g., statistical, spectral measures, etc.).

$\{\bar{f}^{(u)}\}_{u=1}^N$ are partitioned into K clusters $\{C_k\}_{k=1}^K$ using an unsupervised algorithm (e.g., k-means), with centroids $\{\mu_k\}_{k=1}^K$ as in [8]. For each cluster C_k , a separate GCN model $g_\theta^{(k)}$ is trained for the users that belong to that group using graph-structured inputs.

Each feature vector f is organized into a tree $T = (V, E)$, where V is the set of nodes and E the set of edges representing parent–child relationships. The leaf nodes $h_i \in \mathbb{R}$ correspond to individual features, while the internal nodes are aggregated representations. The embedding of a parent node h_p is computed using the average value of its children. Repeating this

latter process across modalities or types of signals produces a collection of trees $G = \{T_1, T_2, \dots\}$, which are then converted into graphs. The self-supervised training process aims to predict the parent node of each leaf, i.e., to map $h_i \mapsto p(i)$. The loss function used during training is the cross-entropy loss:

$$\mathcal{L} = - \sum_{i \in \text{leaves}} \log \frac{\exp(z_{i,y_i})}{\sum_{c=1}^C \exp(z_{i,c})}, \quad (1)$$

where C is the number of internal nodes, y_i is the true parent index of leaf i , and $z_{i,c}$ is the logit corresponding to class c .

Edge deployment: The frozen encoders $\{g_\theta^{(k)}\}_{k=1}^K$ and the centroid metadata $\{\mu_k\}_{k=1}^K$ will be deployed into edge. When raw multivariate signals $x^{(u^*)} \in \mathbb{R}^{T \times S}$ from an unseen new user u^* enter the edge, the feature vector $\bar{f}^{(u^*)} \in \mathbb{R}^F$ will be extracted in the same manner as in the pre-deployment phase. The cluster corresponding to the user k^* is computed based on the closet centroid. Once k^* is identified, the edge device downloads the corresponding frozen encoder $g_\theta^{(k^*)}$. The final step consists of constructing a task- and user-specific classifier ϕ_ψ , trained *from scratch* on-device using a few-shot labeled subset $\mathcal{D}_{\text{train}}^{(u^*)} = \{(x_j, y_j)\}_{j=1}^M$ of the new user’s data, where M is the number of labeled samples available for user u^* and y_j denotes the task label (e.g., class membership). The encoder $g_\theta^{(k^*)}$ is kept frozen, and its output embeddings are fed into the user-specific classifier on the edge: $(\hat{y}_j = \phi_\psi(g_\theta^{(k^*)}(x_j)))$. Training of the task-specific classifier is performed under cross-entropy loss:

$$\mathcal{L}_{\text{edge}} = - \sum_{(x_j, y_j) \in \mathcal{D}_{\text{train}}^{(u^*)}} \log p_{\psi, \theta}(y_j | x_j), \quad (2)$$

where $p_{\psi, \theta}(y_j | x_j)$ is the probability assigned by the combined encoder–classifier model to label y_j for input x_j .

III. EXPERIMENTAL RESULTS

Datasets: Two multi-modal physiological datasets were employed: WEMAC [9] and WESAD [10]. Both contain synchronized recordings of three main physiological modalities: blood volume pulse (BVP), galvanic skin response (GSR), and skin temperature (SKT). WEMAC has two labels: fear and non-fear. WESAD has three labels: baseline, stress and amusement.

Experimental setup: From all the modalities, a total of $F = 123$ descriptors or features were derived: 84 from BVP, 26 from GSR, and 13 from SKT. All feature vectors $\{\bar{f}^{(u)}\}_{u=1}^N$ were clustered into $K = 3$ groups, i.e., $\{C_k\}_{k=1}^3$, a choice that empirically provided the best balance between capturing inter-subject variability and model simplicity. To ensure robust evaluation, a Leave-One-Subject-Out cross-validation strategy is adopted. The NVIDIA Jetson Orin Nano Developer Kit has been chosen as the deployment platform.

Validation results: For validation, two edge-computing-related baselines from previous studies, [11] and [7], are employed for comparison with CHEER in Table I. For WEMAC, although CHEER-Cluster performs slightly below related works, it achieves this using self-supervised GCN training. In contrast, [7] requires fully supervised cloud training with

extensive labels. Additionally, CHEER shows lower standard deviation, indicating greater robustness. For WESAD, despite using fewer labels, CHEER still surpasses [11] in performance.

TABLE I
ACCURACY AND F1-SCORE RESULTS OF CHEER FOR DIFFERENT DL METHODS.

Method	Model	ACC \pm STD	F1 \pm STD
WEMAC			
[7]	CNN-LSTM	79.03 \pm 4.10	78.48 \pm 4.76
CHEER	GCN-Classifier	76.36 \pm 2.58	74.41 \pm 1.84
CHEER-Cluster	GCN-Classifier	78.19 \pm 1.14	77.45 \pm 1.19
WESAD			
[11]	CNN-LSTM	78.91 \pm -	77.73 \pm -
CHEER	GCN-Classifier	78.29 \pm 1.92	77.08 \pm 2.14
CHEER-Cluster	GCN-Classifier	80.08 \pm 2.06	81.62 \pm 1.86

Consumption measurement: In addition to accuracy, this experiment evaluates model size, peak RAM usage, training and inference execution time, power consumption during execution, and overall energy consumption, to show the superior in deployment. The detailed results are presented in Table II and Table III. The results indicate that CHEER significantly reduces the size of the DL model, peak RAM, time and power consumption based on the same manner of edge.

TABLE II
SIZE AND TIME CONSUMPTION OF CHEER DEPLOYMENT.

Method	Size (MB)	Peak RAM (MB)		Time (ms)	
		Train	Infer	Train	Infer
[7]	2.54	10.18	3.82	785.21	239.70
[7] (w/ Jetson)	2.54	90.18	37.58	317.35	89.60
CHEER	0.99	40.08	18.23	87.14	43.88

TABLE III
POWER AND ENERGY CONSUMPTION OF CHEER DEPLOYMENT.

Method	Power (W)		Energy (J)	
	Train	Infer	Train	Infer
[7]	1.02	0.67	0.80	0.16
[7] (w/ Jetson)	5.36	5.36	1.70	0.47
CHEER	1.42	1.42	0.12	0.06

IV. CONCLUSION

This paper introduces CHEER, a framework addressing major cloud–edge AI challenges such as limited labels, high communication cost, and privacy concerns. Tested on the Nvidia Jetson Orin Nano for emotion recognition, CHEER achieves higher accuracy than prior works while greatly reducing model size, latency, and energy use, demonstrating strong efficiency and suitability for real-world edge deployment.

ACKNOWLEDGMENT

This work has been supported by the MANIAS project (PID2023-150922NA-I00) funded by MICIU/AEI/10.13039/501100011033 and by ERDF, EU.; by the Co-HIPSTER-OE project (PID2024-158251NA-C33) funded by MICIU/AEI /10.13039/501100011033 and by ERDF, EU.; by the ETSII of UPM under the project ETSII-UPM25-PU01; and by the CEI Grants Program of UPM.

REFERENCES

- [1] F. Palumbo, M. K. Zedda, T. Fanni, A. Bagnato, L. Castello, J. Castrillon, R. D. Ponte, Y. Deng, B. Driessen, M. Fadda *et al.*, “Myrtus: Multi-layer 360 dynamic orchestration and interoperable design environment for compute-continuum systems,” in *Proceedings of the 21st ACM International Conference on Computing Frontiers: Workshops and Special Sessions*, 2024, pp. 101–106.
- [2] W. Weng, Y. Gu, S. Guo, Y. Ma, Z. Yang, Y. Liu, and Y. Chen, “Self-supervised learning for electroencephalogram: A systematic survey,” *ACM Computing Surveys*, 2024.
- [3] L. Jing and Y. Tian, “Self-supervised visual feature learning with deep neural networks: A survey,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 11, pp. 4037–4058, 2020.
- [4] T.-H. Hsu, Z.-H. Wang, and A. R. See, “A cloud-edge-smart iot architecture for speeding up the deployment of neural network models with transfer learning techniques,” *Electronics*, vol. 11, no. 14, p. 2255, 2022.
- [5] K. Cao, Y. Liu, G. Meng, and Q. Sun, “An overview on edge computing research,” *IEEE access*, vol. 8, pp. 85 714–85 728, 2020.
- [6] A. Alwarafy, K. A. Al-Thelaya, M. Abdallah, J. Schneider, and M. Hamdi, “A survey on security and privacy issues in edge-computing-assisted internet of things,” *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4004–4022, 2020.
- [7] J. Sun, L. G. Martín, C. L. Ongil, J. Miranda, J. Portilla, and A. Otero, “Solving the cold-start problem for the edge: Clustering and adaptive deep learning for emotion detection,” in *2025 Design, Automation & Test in Europe Conference (DATE)*. IEEE, 2025, pp. 1–7.
- [8] L. Gutiérrez-Martín, C. López-Ongil, J. M. Lanza-Gutiérrez, and J. A. Miranda Calero, “Personalized clustering for emotion recognition improvement,” *Sensors*, vol. 24, no. 24, p. 8110, 2024.
- [9] J. A. Miranda, E. Rituerto-González, L. Gutiérrez-Martín, C. Luis-Minguez, M. F. Canabal, A. R. Bárcenas, J. M. Lanza-Gutiérrez, C. Peláez-Moreno, and C. López-Ongil, “Wemac: Women and emotion multi-modal affective computing dataset,” *arXiv preprint arXiv:2203.00456*, 2022.
- [10] P. Schmidt, A. Reiss, R. Duerichen, C. Marberger, and K. Van Laerhoven, “Introducing wesad, a multimodal dataset for wearable stress and affect detection,” in *Proceedings of the 20th ACM International Conference on Multimodal Interaction*, ser. ICMI ’18, 2018, p. 400–408. [Online]. Available: <https://doi.org/10.1145/3242969.3242985>
- [11] J. Sun, J. Portilla, and A. Otero, “Negative emotion recognition based on physiological signals using a cnn-lstm model,” *2024 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 3736–3741, 2024.