2026 IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob 2026)
August 1-4, 2026. Edmonton, Canada

Real-Time Fall Detection for Care Recipient in Ambient Assisted Living Using Skeleton-Based LSTM on Edge Devices

H. Haghjoo[1], R. Heidari[1], M. Tavakoli[2], and H. D. Taghirad[1]

***Abstract*— Falls represent a critical biomedical challenge for aging populations, often leading to severe injury or long-life scenarios if assistance is delayed. While fall prevention strategies focus on gait analysis, effective post-impact fall detection is essential for ensuring timely emergency escalation. Ambient vision-based systems offer a compelling alternative to wearable sensors by enabling reliable, passive monitoring without compliance burdens; however, widespread deployment faces significant hurdles regarding privacy, scalability, and system reliability. This paper proposes SOLA: Skeleton-based Optimized LSTM for Action recognition, a resource-efficient, privacy-centric pipeline designed for local execution in multi-camera environments. Rather than relying on cloud processing or expensive dedicated workstations per room, SOLA optimizes the trade-off between computational demand and classification precision to enable robust operation on cost-effective local hardware. By synergizing skeletal pose estimation with an evolutionary-optimized LSTM network, the architecture minimizes false alarms, a crucial requirement for dependable emergency alerting. Furthermore, the lightweight design ensures that the system remains operational during network outages or partial infrastructure failures, prioritizing safety in worst-case scenarios. Experimental results on the GMDCSA-24 dataset demonstrate that SOLA achieves superior inference speed and low false-positive rates, offering a scalable solution for privacy-preserving, decentralized home monitoring.**

I. Introduction

The world is on the threshold of a profound demographic transformation, with the population aged 65 or older expected to reach 2.1 billion by 2050 [1]. This demographic shift creates an urgent demand for assistive technologies (AT) to support the growing number of older individuals. Human Activity Recognition (HAR), which uses sensor data and artificial intelligence to interpret human behavior, stands as a cornerstone of modern AT. However, the adoption of traditional HAR systems faces significant user-centric and technical hurdles that often outweigh their functional benefits. For instance, many older adults associate visible medical devices with frailty, leading to feelings of stigma. When combined with physical discomfort and the maintenance burden of charging, this results in poor adherence, rendering the devices useless when removed [2]. In contrast, vision-based systems powered by pose estimation offer a compelling alternative. By extracting specific kinematic and geometric features from a person's digital skeleton, these systems enable the objective, continuous, and passive assessment of complex Activities of Daily Living (ADLs) [3]. This approach further facilitates the quantification of motor symptoms in neurodegenerative diseases and allows for proactive fall prediction, potentially revolutionizing in-home care.

AT, particularly within the framework of Ambient Assisted Living (AAL), leverages information and communication technologies to enhance the quality of life, safety, and independence of individuals, especially older adults and people with disabilities [4]. The primary objective of AAL is to foster intelligent environments that support users in their daily routines, empowering them to live autonomously while ensuring assistance is readily available when needed [5].

Assistive robots serve as a prominent example of such systems. However, their costs vary dramatically, ranging from approximately $700 for the first year of an ElliQ social robot to $5,500 for a PARO therapeutic robot. Professional systems can be even more expensive, estimated at $85,000 per year, a cost potentially exceeding that of a human caregiver [6]. To address these cost barriers and privacy concerns, alternative solutions may include deploying lightweight vision-based models running over smart home camera environments and processed locally on edge devices, such as a System-on-Chip (SoC), to minimize or eliminate the need to transmit raw video to the cloud.

II. Related Work

A. Volumetric and Spatiotemporal Methods

HAR has traditionally utilized volumetric approaches that directly learn spatiotemporal features from raw video frames. The Inflated 3D ConvNet (I3D) established a paradigm shift by expanding 2D classification kernels into 3D, allowing for the end-to-end learning of temporal coherence [7]. While I3D achieved benchmark-setting performance on datasets like UCF-101, its high computational requirements often hinder deployment on edge devices. Furthermore, separable 3D convolutions (S3D) reduce parameter counts by decomposing volumetric operations into sequential 2D spatial and 1D temporal steps, which is critical for low-latency assistive technologies [8], [9].

B. Skeleton-based and Geometric Representations

Skeleton-based methods reduce high-dimensional visual data to sparse human body topologies, offering intrinsic advantages for real-time monitoring [10]. These methods typically utilize hand-crafted geometric descriptors, such as

[1]Hossein Haghjoo, Reza Heidari and Hamid D. Taghirad are with the Applied Robotics & AI Solutions (ARAS), Faculty of Electrical and Computer Engineering, Industrial Control Center of Excellence, K. N. Toosi University of Technology, Tehran, 19697-64499, Iran. Corresponding Author's Email: taghirad@kntu.ac.ir

[2]Mahdi Tavakoli is with the Department of Electrical and Computer Engineering and Department of Biomedical Engineering, University of Alberta, Edmonton, AB, Canada, T6G 1H9

joint-to-joint distances, which provide invariant representations independent of camera viewpoint [11]. Advancements in learned representations, particularly via Graph Convolutional Networks (GCNs), have enabled models like ST-GCN to discover task-specific joint relationships directly from data [12]. Modern adaptive variants like DeGCN and AGMS-GCN further enhance performance by learning dynamic graph topologies [13]. For edge deployment, modular pipelines often utilize a two-stage approach: a lightweight pose extractor followed by temporal aggregation through LSTM or GRU cells to balance accuracy and power budgets [14].

C. Temporal Modeling and Motion Descriptors

For long-range temporal modeling, Temporal Segment Networks (TSN) utilize sparse sampling to capture video-level information without the burden of dense frame processing [15]. Similarly, SlowFast networks employ a biologically inspired design with dual pathways at different temporal resolutions to separate semantic content from rapid motion [16]. These frameworks are increasingly integrated with model compression and pruning techniques to facilitate high-performance inference on wearable sensors and embedded ARM processors [17].

D. Assistive Technologies

The convergence of skeletal recognition and efficient architectures has enabled privacy-preserving assistive technologies for elderly populations. Real-time systems now integrate diverse models like TimeSformer and UniFormerV2 to predict functional decline and emergency events, such as chest pain or staggering, within smart home infrastructures [18]. These applications prioritize high sensitivity and on-device personalization to adapt to the evolving physical capabilities of individual users.

In this work, a comprehensive pipeline is proposed for skeleton-based fall detection in ambient assisted living environments, utilizing Long Short-Term Memory (LSTM) networks. Given the high dimensionality of the hyperparameter search space, a Genetic Algorithm (GA) is employed for optimization, thereby maximizing detection sensitivity while minimizing false alarm rates to the lowest feasible extent. To ensure computational efficiency and a lightweight architecture, quantization techniques and optimization frameworks (e.g., OpenVINO, ONNX) are utilized for the person detection module. Furthermore, to mitigate excessive architectural complexity within the LSTM layers, a strategic trade-off is established among pattern discrimination capability, inference latency, and false-positive tolerance.

The contributions of this paper are as follows:

1) A hardware-agnostic, lightweight, and privacy-preserving fall-detection pipeline engineered on constrained computational resources.
2) An adaptive optimization framework that utilizes GA and data science techniques to tune parameters for environmental robustness.

The remainder of this paper is structured as follows. Section III details the proposed architecture and describes the dataset utilized for training and validation. Section IV presents a comprehensive evaluation of the model across various experimental scenarios. Finally, Sec. V summarizes the key findings and discusses the broader implications of this work.

III. Proposed SOLA Framework

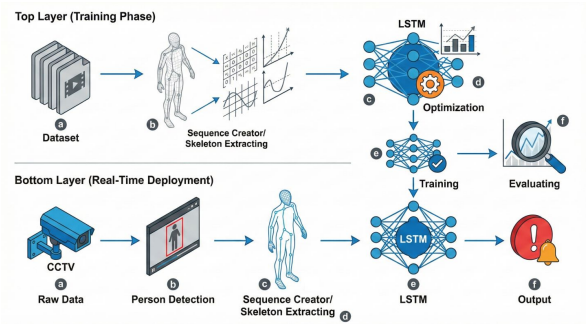

Fig. 1. Overview of the proposed Skeleton-Based Optimized LSTM for Action (SOLA) framework. The architecture is stratified into two operational phases: (Top) The Training Phase, where the dataset (a) undergoes standardization (b) and evolutionary hyperparameter optimization via a GA (c, d) to derive the optimal model configuration (e) prior to evaluation (f). (Bottom) The Real-Time Deployment Phase, designed for edge devices, which processes raw input (a) through lightweight person detection (b) and skeletal feature extraction (c, d) to trigger immediate fall alerts (f) using the optimized LSTM model.

A. Human Fall Dataset

To effectively address the recognition of human activities based on skeletal data in ambient assistive living scenarios, it is crucial to select a dataset that encompasses accurate patterns. After careful consideration, we opted for the GMDCSA-24 dataset [19]. Published in 2024, this dataset specifically targets the challenges of fall detection in low-resource environments. It comprises 160 video sequences (81 falls and 79 ADL (labeled as normal)) performed by four subjects across three distinct home environments. Distinctively, GMDCSA-24 employs a low-resolution 0.92 MP webcam to simulate the visual quality of standard surveillance systems, and it incorporates challenging lighting conditions (day and night) as well as complex ADLs (e.g., picking up objects, lying down) to rigorously test the classifier's ability to distinguish actual falls from similar non-fall motions.

B. Training Phase

In this phase, we focus on generating optimal input sequences and calibrating the model parameters using evolutionary algorithms and cost-sensitive objectives.

1) Data Preprocessing and Sequence Generation: To ensure the integrity of the input pipeline, the raw dataset undergoes a strict standardization process. Each video clip is trimmed to a maximum duration of 10 seconds and filtered to contain a single subject performing the target action. We employ a lightweight person detection and tracking module (using YOLO-based models) to crop and resize the region of

interest (ROI) to 224×224 pixels, storing each action class in distinct directories (Fig. 1, Top-a).

Temporal sequences are generated using a sliding window technique. The optimal window length is treated as a hyperparameter and determined via the optimization stage described below. Applying this sliding window across all video frames with an appropriate stride serves as a data augmentation strategy, significantly enhancing model robustness.

2) Skeletal Feature Extraction: Following sequence generation, we utilize MediaPipe [20] to extract 3D skeletal topologies. Unlike earlier iterations of our framework that relied on 2D detectors (e.g., YOLO) yielding 17×3 vectors, which lacked sufficient discriminative power for complex falls, the current pipeline extracts 33 landmarks with 4 dimensions $(x, y, z, \text{visibility})$. This transition to 3D geometry captures intricate patterns and micro-movements essential for high-precision recognition (Fig. 1, Top-b).

To maintain a lightweight architecture suitable for LSTM processing on edge devices, we deliberately avoid auxiliary CNN feature extractors. Instead, we rely on geometric feature engineering. All 33 landmarks are spatially normalized relative to the central hip point to ensure invariance to subject position. When partitioning the data into training (70%, 112 videos), validation (20%, 32 videos), and test sets (10%, 16 videos), it is critical to ensure that sequences from a single event are not distributed across multiple subsets. Preventing this overlap is essential, as it causes data leakage and results in misleading accuracy metrics. Prior to training, a heuristic data cleaning phase is applied: landmarks with low visibility scores or physiologically impossible configurations (e.g., illogical distances between face and hands or excessive frame-to-frame displacement) are zeroed out, and highly corrupted sequences are discarded.

3) Cost-Sensitive Optimization for Critical Events: While the experimental dataset employed in this study maintains a balanced distribution between fall and non-fall sequences, real-world deployment scenarios often exhibit severe class imbalance, where critical events are statistically rare compared to routine ADLs. To ensure our framework generalizes effectively to these high-stakes, imbalanced environments, we integrated a cost-sensitive learning objective. Rather than treating all misclassifications equally, we adopt a Weighted Cross-Entropy (WCE) loss function. This formulation assigns a higher penalization weight to the minority or critical class, thereby preventing the model from converging toward a trivial solution that biases the majority class.

Mathematically, for a batch of N sequences and C classes, the weighted objective function $\mathscr{L}_{WCE}$ minimizes the divergence between the ground truth y and the prediction $\hat{y}$ as follows:

$$\mathscr{L}_{WCE} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} w_c \cdot y_{i,c} \log(\hat{y}_{i,c}) \tag{1}$$

where $y_{i,c}$ denotes the binary indicator (0 or 1) if class c is the correct classification for sample i, and $\hat{y}_{i,c}$ is the predicted probability. The class weight w_c is inversely proportional to the class frequency, computed as $w_c = N_{total}/(C \cdot N_c)$, ensuring that gradients derived from rare, critical events are amplified during backpropagation.

4) Evolutionary Hyperparameter Optimization: Hyperparameter tuning was guided by clinical risk rather than purely algorithmic performance. Falls with delayed assistance can lead to clinically relevant complications (e.g., dehydration, pressure injuries, muscle/tissue damage, delayed treatment, psychological harm), making false negatives a high-severity failure mode. Conversely, excessive false alarms can contribute to alarm fatigue and desensitization, which is a recognized patient-safety hazard that can delay or suppress response to true events. Consistent with benefit–risk thinking that explicitly considers false-positive and false-negative outcomes, we therefore optimized a cost-sensitive objective that prioritizes fall recall while penalizing false alarms through the temporal decision parameters (window length and streak threshold), which directly control alert frequency and time-to-alert.

To navigate the high-dimensional hyperparameter search space $\mathscr{S}$, we employ a GA, a strategy selected for its robustness in avoiding local minima when dealing with correlated variables. The optimization process targets three distinct categories of parameters: architectural, training, and temporal. First, the structural complexity is tuned by varying the number of stacked layers (1 to 3) and the hidden unit dimensions, which range from 64 to 384 in increments of 32. Second, the training dynamics are regulated by optimizing the dropout rate, learning rate, and batch size (Fig. 1, Top-c). Finally, the GA simultaneously refines the temporal logic by adjusting the sliding window sequence length and the "streak threshold," defined as the specific number of consecutive positive frame predictions required to trigger a confirmed fall alert.

The optimization process is iterative. In each trial, a candidate model is trained for a limited epoch budget of 35 epochs and evaluated on a specific data subset, referred to as the *optimization validation set*. We assess performance using metrics such as precision, recall, and F1-score. Based on these results, the GA updates the parameter population, iteratively converging toward the optimal set Θ^* for the target environment. Algorithm 1 illustrates the whole process of optimization (Fig. 1, Top-d).

5) Final Training and Evaluation: Upon convergence of the GA, the final model is retrained using the optimal parameters Θ^* with an extended epoch schedule. This ensures that the final performance meets or exceeds the results obtained during the search phase. The output is a serialized PyTorch model optimized for the inference pipeline (Fig. 1, Top-e). In the final stage, the model undergoes rigorous evaluation to quantify precision, robustness, stability, and generalizability on unseen test data (Fig. 1, Top-f).

The hyperparameter space is explored via a GA on a challenging data subset to identify optimal configurations. Analysis includes both raw sequence-level accuracy from the LSTM output and the final accuracy of the integrated SOLA system, which incorporates post-processing FSM

logic (Sec. III-C.4) to bolster confidence and mitigate false alarms.

Algorithm 1 SOLA Hyperparameter Optimization

Require: $\mathscr{D}_{train}$, $\mathscr{T}_{val}$, Search Space $\mathscr{S}$, Trials N
Ensure: Optimal Parameters Θ^*
$\mathscr{H} \leftarrow \emptyset$
for $i \leftarrow 1$ to N **do**
  $\Theta_i \leftarrow \text{Suggest}(\mathscr{S}, \mathscr{H})$
  $\mathscr{M}_i \leftarrow \text{Train}(\mathscr{D}_{train}, \Theta_i)$ ▷ Train for fixed epochs
  $Y_{true}, Y_{pred} \leftarrow \emptyset, \emptyset$
  for $V \in \mathscr{T}_{val}$ **do** ▷ Evaluation Phase
    $\mathbf{p} \leftarrow \text{Predict}(\mathscr{M}_i, V)$ ▷ Seq. of probabilities
    $\hat{y} \leftarrow 0$
    if $\exists$ sub-sequence $s \subset \mathbf{p} : \forall p \in s, p > 0.5 \wedge |s| \geq \Theta_i.\tau_{\text{streak}}$ **then**
      $\hat{y} \leftarrow 1$ ▷ Fall detected via streak logic
    end if
    $Y_{pred}.\text{append}(\hat{y})$; $Y_{true}.\text{append}(\text{Label}(V))$
  end for
  $score \leftarrow \text{MacroF1}(Y_{true}, Y_{pred})$
  $\mathscr{H} \leftarrow \mathscr{H} \cup \{(\Theta_i, score)\}$
end for
return $\arg\max_{\Theta \in \mathscr{H}}(\text{score})$

C. Deployment Phase

The deployment phase addresses the constraints of edge execution, focusing on concurrency, memory management, and temporal decision logic.

1) Real-Time Inference Challenges: A fundamental distinction between training and live deployment lies in the observability of the temporal sequence. While offline methods benefit from access to complete, holistic video sequences, real-time inference is inherently causal; decisions must be derived solely from partial sequences observed up to the current timestamp. Furthermore, the deployment environment introduces the complexity of multi-person scenarios, necessitating a robust pipeline to isolate and track individuals independently.

2) Multi-Stage Preprocessing Pipeline: To address these challenges, we implement a modularized preprocessing pipeline. First, we employ a YOLO-based object detector [21] to identify human subjects within the frame. To enhance reliability in varying conditions, the detector can be either pre-trained or fine-tuned on environment-specific samples. Both approaches were evaluated, and the pre-trained model was ultimately selected for its generalization performance. Upon detection, each subject is isolated via an ROI crop and resized to standard dimensions of 224×224 pixels (Fig. 1, Bottom-b). Since each cropped region contains a single subject, we invoke the MediaPipe framework to extract 33-point skeletal landmarks.

To ensure consistency with the training distribution, we apply the same spatial normalization protocols used during the training phase (Fig. 1, Bottom-c). Furthermore, to mitigate sensor noise and accurately capture high-velocity dynamics, such as rapid limb movements during a fall, we implement a cascaded filtering approach. A Kalman filter is combined with a Butterworth low-pass filter to smooth the skeletal trajectories, ensuring that even subtle micro-movements are preserved for the downstream classifier (Fig. 1, Bottom-d).

3) Hybrid Concurrency Architecture: Deploying this pipeline across a multi-camera smart home network necessitates an efficient concurrency model. We evaluated both multi-threading and multiprocessing paradigms. While threading offers a lightweight memory footprint, it suffers from significant contention due to the Global Interpreter Lock (GIL) in Python-based environments. This contention resulted in variable frame processing rates and out-of-order execution, causing the model to miss critical frames essential for action recognition. Conversely, a pure multi-processing approach ensures process isolation but incurs a prohibitive memory overhead, limiting the number of active camera streams on resource-constrained hardware.

To resolve this trade-off, we developed a hybrid architecture. Static, heavy dependencies (e.g., model weights) are loaded once into shared memory, while dynamic stream processing is handled by isolated processes that reference these shared assets via pointers. This design eliminates race conditions and ensures deterministic frame processing while maximizing the number of supportable camera streams. Additionally, to mitigate packet loss inherent to wireless home networks, we enforce RTSP-over-TCP transport protocols, ensuring the reliable delivery of sequential frames and preventing temporal discontinuities that could degrade LSTM performance.

4) Temporal Decision Logic and State Machine: The final classification decision is governed by a sliding window mechanism that aggregates the LSTM's frame-level predictions over a history of 100 sequences. A naive majority vote (averaging) is insufficient for fall detection, as intermittent false negatives can dilute the global score. Instead, we employ a "streak-based" heuristic inspired by peak detection signal processing.

The decision logic is modeled as a Finite State Machine (FSM). Initially, the subject is in a *Normal* state. If the density of positive fall predictions within the window exceeds a preliminary threshold, the system transitions to a *Suspicious* state. A confirmed event is triggered only if a stricter secondary threshold is met and a contiguous "streak" of positive predictions is observed (Fig. 1, Bottom-f). This multi-stage gating mechanism significantly reduces false alarms while maintaining high sensitivity to genuine fall events.

IV. Experiments

In this section, we evaluate the performance of the SOLA framework. We first analyze the dynamics of the evolutionary optimization process to identify the critical determinants of model performance. Subsequently, we examine the training convergence and feature separability of the optimized architecture. Finally, we present the classification metrics on the hold-out test set, with a specific focus on the trade-off between sensitivity and false alarm rates in resource-constrained deployment scenarios.

A. Evolutionary Optimization Dynamics

The high-dimensional nature of the hyperparameter space necessitates an automated search strategy to balance model complexity with inference precision. We executed the genetic optimization process over 50 trials. Fig. 2 illustrates the evolution of the objective function (F1-Score) across these iterations. The optimization landscape proved to be highly non-convex, characterized by significant performance variance between trials. The optimal configuration was identified at iterations 11 and 45, achieving a maximum validation F1-score of 0.7603. The variability observed in subsequent iterations underscores the sensitivity of the model to hyperparameter perturbations and justifies the use of an evolutionary approach over manual tuning.

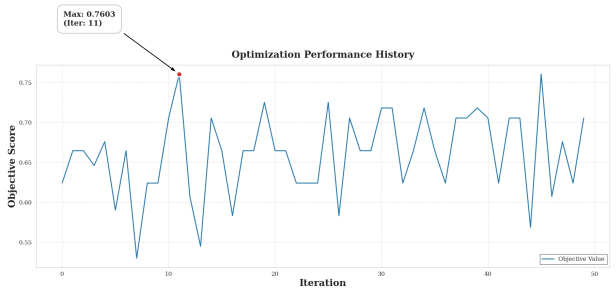

Fig. 2. Evolution of the objective function (F1-Score) across 50 optimization trials.

To understand which parameters most significantly influence detection performance, we conducted a hyperparameter importance analysis using a functional ANOVA approach. A critical finding of this study is the dominance of the temporal decision logic over architectural depth. The `streak_threshold`, the parameter governing the FSM's transition sensitivity, accounted for 50% of the variance in the objective function. In contrast, architectural parameters such as `num_layers` (0.05) and `hidden_size` (0.08) exhibited marginal impact.

In the slice plot analysis (Fig. 3), high-performing clusters are tightly correlated with specific ranges of learning rates and streak thresholds, while architectural parameters show a more uniform distribution of scores.

B. Training Convergence

Following the identification of the optimal hyperparameter set Θ^*, the final model was trained for 120 epochs. The training dynamics, depicted in Fig. 4, demonstrate a stable convergence profile. The training loss decreases monotonically, while the validation accuracy rapidly ascends to a plateau of approximately 98%. Although minor volatility is observed in the validation loss, the absence of significant divergence indicates that the model generalizes well without overfitting to the training noise.

To inspect the learned representations, we visualized the high-dimensional feature embeddings of the penultimate LSTM layer using t-Distributed Stochastic Neighbor Embedding (t-SNE), as shown in Fig. 5. The plot reveals two

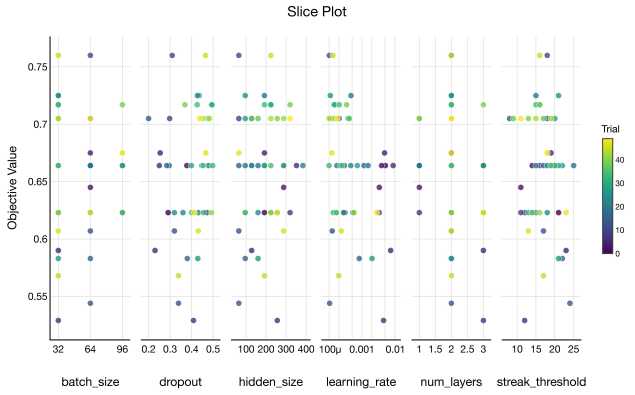

Fig. 3. Slice plot analysis of hyperparameter sensitivity. Each subplot visualizes the marginal relationship between a specific hyperparameter and the objective value.

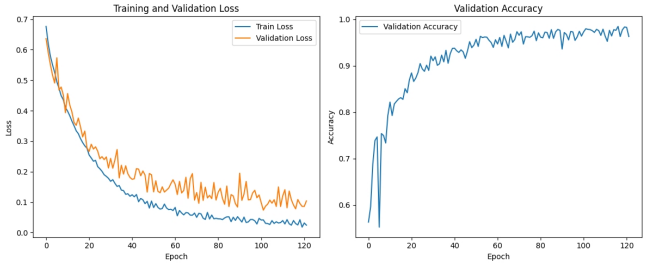

Fig. 4. Training and validation curves for the optimized LSTM model.

distinct global clusters corresponding to 'Fall' and 'Normal' classes. However, a noticeable "transition zone" exists where class boundaries overlap. These overlapping regions likely correspond to ambiguous activities (e.g., rapid lying down or stumbling) that kinematically resemble falls. Critically, the LSTM operates as a feature encoder, not the final classifier. Sequence-level predictions are aggregated through a downstream temporal decision module that employs streak-based logic and an FSM (Sec. III-C.4).

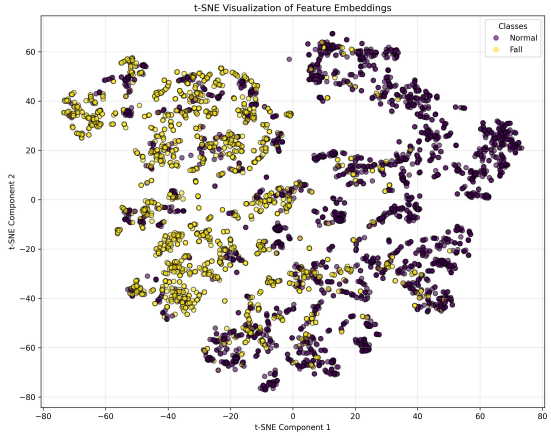

Fig. 5. t-SNE visualization of the feature embeddings extracted from the LSTM's hidden states.

C. Classification Performance and Robustness

The primary objective of the SOLA framework is to provide reliable monitoring without inducing "alert fatigue"

through false alarms. The confusion matrix on the independent test set, presented in Fig. 6, confirms the system's efficacy in high-stakes environments. The system maintained high sensitivity, correctly identifying 2,169 fall events.

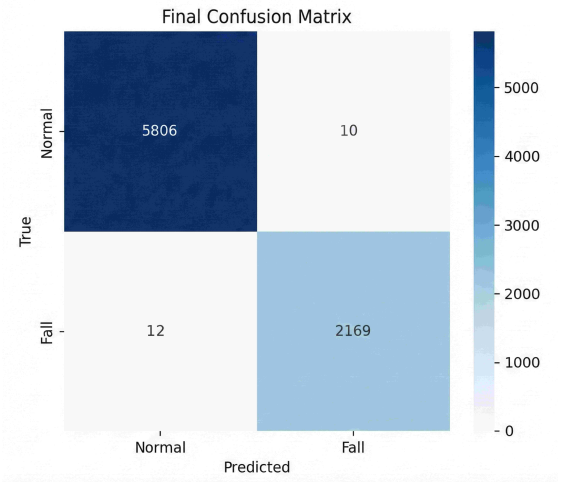

Fig. 6. Confusion matrix for Fall vs. Normal classification on the test set.

To provide deeper insight into the model's real-time decision-making process, Fig. 7 visualizes the inference dynamics over a continuous video sequence containing a fall event. The model exhibits exceptional stability in the *Normal* state (sequences 0–28), maintaining confidence levels near 1.0 with zero fluctuations. This stability is critical for minimizing false positives during long-duration monitoring. Second, the onset of the fall at sequence index 29 is marked by a sharp, decisive transition. Notably, the confidence score exhibits a characteristic dip (decreasing to approximately 0.6) at the transition boundaries (indices 29 and 34). This behavior is analytically consistent with the kinematic ambiguity inherent in the initiation and recovery phases of a fall, where the subject's pose may momentarily share geometric similarities with non-fall activities (e.g., crouching).

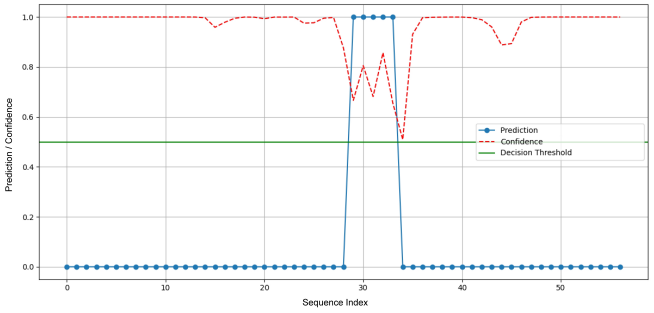

Fig. 7. Temporal inference timeline for a sample test sequence. The blue markers indicate the binary classification state (0: Normal, 1: Fall), while the red dashed line tracks the prediction confidence. The green line denotes the decision threshold (0.5).

D. Comparative Analysis

To validate the proposed framework against recent state-of-the-art benchmarks, we compared SOLA with the algorithms introduced by Yu *et al.* [22], including individual Temporal Convolutional Network (TCN) and Transformer Encoder (TE) models, as well as their hybrid TCN-TE architecture. Table I summarizes the performance metrics on the GMDCSA-24 dataset.

The comparison highlights a distinct trade-off between sensitivity and false alarm suppression. While the reference TCN-TE model achieves the highest sensitivity (86.81%), it incurs a false positive rate of approximately 6.28%. In contrast, SOLA employs a multi-thresholding strategy governed by the "streak" logic described in Sec. III. This restrictive decision boundary deliberately prioritizes precision to mitigate "alert fatigue," resulting in a reduced false positive rate of 4.87% while maintaining a competitive sensitivity of 83.33%, comparable to the standalone TE model (83.35%). Although this conservative design yields a slightly lower accuracy (87.51%), it ensures that the system remains robust against false triggers in long-duration monitoring, a critical prerequisite for autonomous emergency dispatch.

In terms of computational efficiency, SOLA demonstrates superior scalability across diverse hardware tiers. The reference implementations require an NVIDIA Jetson Orin NX (approx. 100 TOPS) to achieve 19–21 FPS for a single camera stream. Conversely, our optimized pipeline maintains viable real-time performance (7 FPS) on a low-power Raspberry Pi 4 (E.1) and scales up to 30 FPS on a high-performance server (E.4). Notably, these throughput figures for SOLA account for a hybrid concurrency architecture capable of handling multiple input camera streams simultaneously, whereas the reference benchmarks are reported for a single-camera input.

E. Deployment Benchmarks

To evaluate the feasibility of the SOLA framework for real-world ambient monitoring, we conducted comprehensive benchmarking on three distinct edge devices representing different tiers of computational capability, alongside a high-performance non-edge baseline. The hardware configurations tested are as follows:

E.1 (SBC): Raspberry Pi 4B (Cortex-A72 @ 1.5 GHz, 4 GB LPDDR4). Represents a power-constrained entry-level edge node.

E.2 (Mini PC): Intel i5-7200U (2C/4T, up to 3.1 GHz, 8 GB DDR4). Simulates a standard home gateway or media server.

E.3 (Desktop): Intel i5-11400 (6C/12T, 16 GB DDR4) with NVIDIA GTX 1650 (4 GB VRAM). Represents a consumer-grade workstation.

Cloud (Server): Intel i9-13900K, 64 GB DDR5, and NVIDIA RTX 4090 (24 GB VRAM). A high-performance baseline for unconstrained centralized inference.

All benchmarks measure the complete end-to-end pipeline described in Section III-C. We present four key metrics in Table II: throughput (measured in Frames Per Second), peak application memory usage, average CPU/GPU utilization (expressed as a percentage), maximum number of cameras supported, and cost. These metrics detail the total number of cameras that can be tolerated by each hardware configuration.

TABLE I

PERFORMANCE COMPARISON ON GMDCSA-24 DATASET

	Reference Models [22]			Proposed Model
Metric	**TCN**	**TE**	**TCN-TE**	**SOLA**
Sensitivity (Recall)	81.33% ± 6.23	83.35% ± 2.61	86.81% ± 4.20	83.33% ± 4.3
Accuracy	91.62% ± 3.82	90.99% ± 3.12	92.99% ± 2.50	87.51% ± 2.2
False Positives	≈ 7.15%	≈ 8.11%	≈ 6.28%	≈ 4.87%
System FPS[1]	20 FPS	21 FPS	19 FPS	7 / 15 / 21 / 30 FPS

[1] Reference models measured on NVIDIA Jetson Orin NX; SOLA measured on E.1 / E.2 / E.3 / E.4, described in Sec. IV-E.

V. CONCLUSIONS

This work presented SOLA, a resource-efficient, privacy-preserving framework for real-time fall detection in ambient environments. By synergizing lightweight skeletal pose estimation with an optimized LSTM architecture, we addressed the critical trade-off between computational latency and classification precision inherent to edge-based deployment. The automated pipeline supports model adaptation to real-world falls through incremental retraining on enriched datasets. No cumulative latency beyond task-inherent delays was observed, confirming scalability for continuous monitoring. Crucially, by processing data locally on SoC hardware, the proposed pipeline preserves user privacy using a skeletal-based approach, a prerequisite for the ethical adoption of surveillance technologies in healthcare. Future work will focus on expanding the application to other important HARs. Ultimately, these advancements will contribute to the development of more accessible and proactive ambient assisted living systems capable of bridging the gap between technological capability and user acceptance.

TABLE II

DEPLOYMENT BENCHMARKS.

Metric	**E.1**	**E.2**	**E.3**	**Cloud**
Throughput (FPS)	7	15	21	30
Peak memory (GB)	4	7	12	50
CPU / GPU util. (%)	95 / –	80 / –	25 / 80	20 / 75
Max. input cameras	1	3	5	15
Cost tier	Low	Med.	Med.	High

REFERENCES

[1] W. H. Organization and U. N. C. Fund, *Global report on assistive technology*. World Health Organization, 2022.

[2] A. Marakhimov and J. Joo, "Consumer adaptation and infusion of wearable devices for healthcare," *Comput. Hum. Behav.*, vol. 76, pp. 135–148, 2017.

[3] N. Jlidi, S. Kouni, O. Jemai, and T. Bouchrika, "Mediapipe with gnn for human activity recognition." *J. Universal Comput. Sci. (JUCS)*, vol. 30, no. 6, 2024.

[4] S. Blackman, C. Matlo, C. Bobrovitskiy, A. Waldoch, M. L. Fang, P. Jackson, A. Mihailidis, L. Nygård, A. Astell, and A. Sixsmith, "Ambient assisted living technologies for aging well: a scoping review," *J. Intell. Syst.*, vol. 25, no. 1, pp. 55–69, 2016.

[5] T. Kleinberger, M. Becker, E. Ras, A. Holzinger, and P. Müller, "Ambient intelligence in assisted living: enable elderly people to handle future interfaces," in *Int. Conf. Universal Access Hum.-Comput. Interact.* Springer, 2007, pp. 103–112.

[6] B. Sawik, S. Tobis, E. Baum, A. Suwalska, S. Kropińska, K. Stachnik, E. Pérez-Bernabeu, M. Cildoz, A. Agustin, and K. Wieczorowska-Tobis, "Robots for elderly care: review, multi-criteria optimization model and qualitative case study," in *Healthcare*, vol. 11, no. 9. MDPI, 2023, p. 1286.

[7] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 6299–6308.

[8] S. B. Hong, Y. H. Kim, S. H. Nam, and K. R. Park, "S3d: Squeeze and excitation 3d convolutional neural networks for a fall detection system," *Mathematics*, vol. 10, no. 3, p. 328, 2022.

[9] H. Wang, J. Lin, and Z. Wang, "Design light-weight 3d convolutional networks for video recognition temporal residual, fully separable block, and fast algorithm," *arXiv preprint arXiv:1905.13388*, 2019.

[10] S. Zhang, X. Liu, and J. Xiao, "On geometric features for skeleton-based action recognition using multilayer lstm networks," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV), 2017*. IEEE, 2017, pp. 148–157.

[11] S. Zhang, Y. Yang, J. Xiao, X. Liu, Y. Yang, D. Xie, and Y. Zhuang, "Fusing geometric features for skeleton-based action recognition using multilayer lstm networks," *IEEE Trans. Multimedia*, vol. 20, no. 9, pp. 2330–2343, 2018.

[12] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Proc. AAAI Conf. Artif. Intell.*, vol. 32, no. 1, 2018.

[13] W. Myung, N. Su, J.-H. Xue, and G. Wang, "Degcn: Deformable graph convolutional networks for skeleton-based action recognition," *IEEE Trans. Image Process.*, vol. 33, pp. 2477–2490, 2024.

[14] S. Campanella, A. Alnasef, L. Falaschetti, A. Belli, P. Pierleoni, and L. Palma, "A novel embedded deep learning wearable sensor for fall detection," *IEEE Sens. J.*, vol. 24, no. 9, pp. 15 219–15 229, 2024.

[15] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks for action recognition in videos," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 11, pp. 2740–2755, 2018.

[16] B. Yang, F. Xin, Z. Wang, and X. He, "High-accuracy detection method for miner action recognition with shuffle attention slowfast networks," *J. Electron. Imaging*, vol. 34, no. 1, pp. 013 012–013 012, 2025.

[17] J. Wang, Z. Li, B. Liu, H. Cai, M. Saada, and Q. Meng, "High-performance inference graph convolutional networks for skeleton-based action recognition," *Neurocomputing*, p. 131078, 2025.

[18] Y. Wang, C. Muntean, P. Pathak, and P. Stynes, "A real-time human action recognition model for assisted living," in *Int. Conf. Eng. Appl. Neural Netw.* Springer, 2025, pp. 3–16.

[19] E. Alam, "ekramalam/GMDCSA24-A-Dataset-for-Human-Fall-Detection-in-Videos: 2.0," Jul. 2024. [Online]. Available: https://doi.org/10.5281/zenodo.12921216

[20] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. G. Yong, J. Lee *et al.*, "Mediapipe: A framework for building perception pipelines," *arXiv preprint arXiv:1906.08172*, 2019.

[21] G. Jocher and J. Qiu, "Ultralytics yolo11," 2024. [Online]. Available: https://github.com/ultralytics/ultralytics

[22] X. Yu, C. Wang, W. Wu, and S. Xiong, "A real-time skeleton-based fall detection algorithm based on temporal convolutional networks and transformer encoder," *Pervasive Mob. Comput.*, vol. 107, p. 102016, 2025.