A Pre-impact Fall Algorithm Based on a Lightweight Re-Parameters-Parallel Convolutional-TCN

Julong PAN [1], Hongyu WANG [1], Jiyi XU [1], Hai Feng XU [2]*

[1] College of Information Engineering, China Jiliang University, 310018, Hangzhou, Zhejiang, China
[2] School of Electronic Information, LiShui Vocational & Technical College, 323000, LiShui, Zhejiang, China

pjl@cjlu.edu.cn, p23030854094@cjlu.edu.cn, 1800303218@cjlu.edu.cn, 3382462@qq.com

Submitted August 24, 2025 / Accepted December 19, 2025 / Online first January 20, 2026

Abstract. *With the aging society intensifying, the problem of elderly falls has become a key issue of social concern. Research on fall prediction based on Internet of Things (IoT) technology has received widespread attention. To effectively predict fall events, a lightweight IoT-based fall prediction model called lwRPPC-TCN (lightweight Re-Parameters-Parallel-Convolutional Temporal Convolutional Network) is proposed. The model utilizes the temporal data collected by IoT sensors in the input stage and achieves efficient decoupled extraction of temporal and spatial features through lwRPPC blocks. The subsequent Temporal Convolutional Networks (TCNs) further strengthens the ability of modeling the global temporal dependency, thus optimizing the processing capability of sensor time-series data. To validate the generalization ability of the model and mitigate fall data scarcity, two public datasets, SisFall and KFall, are fused, and the performance of the model is evaluated by five-fold cross-validation. In addition, a homogeneous (models belong to the same model family) knowledge distillation technique is introduced to improve the performance of the model. Experimental results demonstrate that the proposed lwRPPC-TCN achieves an accuracy of 98.88% on the fused dataset, outperforming existing fall prediction models, with a fall prediction lead time (interval between the fall prediction time and the collision time) of 250 ms, and a compact model size of 60 KB, which makes it suitable and possible to deploy in a resource-constrained wearable device.*

Keywords

Deep learning, fall prediction, knowledge distillation, Re-Parameters-Parallel-Convolutional-Temporal Convolutional Network, inertial sensor, spatio-temporal feature decoupling

1. Introduction

With the global aging population intensifying, fall events have emerged as a predominant cause of accidental injuries and mortality in older adults. According to the World Health Organization, vulnerable fractures are mostly triggered by minor external forces (falling from a standing height). This phenomenon is more likely to occur in older adults. By 2019, data reveal 178 million incident fractures worldwide, a 33.4% increase in the absolute number of new fractures compared to 1990 [1]. This phenomenon not only inflicts severe physical and psychological consequences on the elderly but also imposes substantial socio-economic and caregiving burdens on families and healthcare systems. Therefore, developing smart healthcare systems that can predict fall alarms will become critical to mitigating risks and enhancing safety for aging populations.

Conventional fall protection systems predominantly depend on video surveillance, ground-based sensors, or pressure-sensitive mats [2]. However, these approaches are limited by reliance on fixed-location infrastructure, high implementation costs, and cumbersome deployment. In recent years, with the development of wearable technology and Internet of Things (IoT) devices, fall protection systems based on wearable devices have become a promising alternative. IoT's inertial sensors, such as accelerometers and gyroscopes, are used widely to capture kinematic parameters, facilitate real-time monitoring of postural transitions, and gait dynamics. This multimodal sensor data enables robust fall risk assessment through quantitative analysis of movement patterns preceding instability events.

Nevertheless, deploying efficient and accurate fall prediction models on resource-constrained wearable devices remains challenging. While deep learning, particularly Convolutional Neural Networks (CNNs), has advanced temporal analysis and anomaly detection, conventional CNNs are often computationally intensive and parameter-heavy, rendering them impractical for edge devices with restricted processing capabilities and power budgets. To bridge this gap, lightweight CNN variants (e.g., MobileNet [3], ConvNeXt [4]) had emerged as a critical research focus. These architectures reduce some level of computational, and memory demands through techniques such as Depthwise Separable convolutions and parameter pruning, while preserving almost the same performance, thereby aligning with the constraints of wearable platforms. Build-

DOI: 10.13164/re.2026.0084

ing on these advancements, a novel lightweight fall prediction framework is proposed that it synergizes the hierarchical design of ConvNeXt [4], ModernTCN [5], with the grouping design of GroupNet [6] and combines TCNs [7]. The main contributions are summarized as follows:

(1) A lightweight Re-Parameters-Parallel Convolutional-Temporal Convolutional Network (called lwRPPC-TCN) for fall prediction is proposed, which delivers better performance by decoupling the extraction of temporal and spatial features and modeling global temporal dependency while meeting the deployment requirements of resource-constrained wearable devices and IoT environments.

(2) The lwRPPC-TCN demonstrates scalable performance through modular hyperparameter adjustments. Based on this, the large model lwRPPC-TCN-L and the small model lwRPPC-TCN-S, suitable for deployment, are designed, and the knowledge distillation [8] technique is adopted to effectively improve the performance of the lwRPPC-TCN-S without increasing its parameters.

(3) Evaluations on two public datasets, SisFall [9] and KFall [10], demonstrate that the proposed model exhibits good performance. To enhance its generalization and robustness across diverse real-world scenarios and to address the limitations posed by the scarcity of fall samples in a single dataset, an innovative approach by fusing SisFall and KFall through advanced data preprocessing techniques is proposed.

2. Related Work

2.1 Fall Prediction Methods

The rapid advancement of deep learning has led to numerous fall prediction algorithms, primarily utilizing CNNs and Recurrent Neural Networks (RNNs), particularly Long Short-Term Memory (LSTM) [11]. However, standalone CNNs or RNNs often struggle to comprehensively capture spatio-temporal relationships in motion sensor data. To address this limitation, Koo et al. [12] and Yu et al. [13] proposed a network called Conv-LSTM to combine CNNs and LSTMs, effectively extracting long-term and short-term temporal dependencies in the time-series data. Their approach not only improved temporal efficiency by using CNNs to preprocess input data for LSTMs but also achieved great performance on the two public datasets of SisFall and KFall.

Recent research have explored more sophisticated architectures, including Transformer [14] and Graph Neural Networks (GNNs) [15], to enhance the model's performance. Al-qaness et al. [16] proposed a multi-branch PCNN-Transformer Network that leverages both parallel convolutional pathways and Transformer mechanisms, achieving superior results on the SisFall, UniMib-SHAR, and MobiAct datasets. Wang et al. [17] designed a spatio-temporal graph neural network (SPGN) that reformulates inertial measurement unit (IMU) time-series data as topological graphs to capture spatial features. While demonstrating high specificity on SisFall, DOFDA, and ConFall12 datasets, their approach exhibited some limitations in the sensitivity of fall events.

Although existing methods perform well on benchmark datasets, their high computational complexity and substantial number of parameters constrain their deployment on resource-constrained wearable devices. In contrast, the proposed lwRPPC-TCN model introduces a novel approach by decoupling temporal and spatial feature extraction, enhancing fall prediction accuracy while maintaining model scalability and compactness suitable for wearable devices. Compared to the existing CNN-based models, our approach integrates TCNs to model global temporal dependencies, addressing the challenge of temporal and spatial feature fusion in fall prediction.

2.2 Lightweight Fall Prediction Models

To employ deep learning technology on wearable devices and mobile terminals, the lightweight model design should become a critical research focus, current approaches fall into two categories: one is lightweight network design, Yu et al. [18] proposed a network called TinyCNN, a two-stage quantized network with 546 parameters, achieving an inference time of 0.037 s on an Arduino Nano BLE 33 sense microcontroller; another one is post-hoc compression of existing models, employing techniques like model pruning [19], knowledge distillation [8], [20], and quantization [21]. For instance, Chi et al. [22] implemented a Vision Transformer (ViT)-to-CNN distillation framework. Chi's method improved several performance metrics on the KFall dataset, but the sensitivity was only 94.79%, revealing fundamental limitations in the capacity of the student model to learn discriminative spatio-temporal patterns from limited sensor data. Although the aforementioned models satisfy the requirements for embedded deployment in terms of model size, their predictive performance is suboptimal.

The lwRPPC-TCN exhibits good architectural scalability, where parametric scaling proportionally enhances performance. To optimize the performance, homogeneous knowledge distillation is implemented by training two structurally homogeneous variants: a high-capacity teacher model and a deployable student model. As theorized by Hao et al. [23], homogeneous models share similar inductive biases, enabling more consistent feature space alignment and efficient knowledge transfer. Experiments demonstrate that using knowledge distillation, the lwRPPC-TCN-S model outperforms the lightweight model trained separately in terms of performance while maintaining the model size.

2.3 Data Enhancement and Data Fusion

Beyond architectural challenges, the inherent rarity of fall events in the real world poses significant data acquisi-

tion hurdles. Data augmentation methods such as flipping, rotating, cropping, scaling, etc., commonly used in the computer vision field, do not apply to multivariate time-series data due to their disruption of temporal causality. While generative adversarial networks (GANs) [24] have been explored for data generation, it is not possible to prove whether the data generated by GANs is fall or Activity of Daily Living (ADL).

While most prior research focuses on individual datasets, a novel approach for constructing a comprehensive training dataset by fusing two publicly available datasets, SisFall and KFall, is proposed to overcome challenges related to sensor heterogeneity, temporal misalignment, and data imbalance through advanced data preprocessing techniques. This dataset fusion enhances the representativeness of the training data, enabling more robust and generalized fall prediction across diverse environments and populations. Experimental results demonstrate that the proposed lwRPPC-TCN model exhibits good prediction performance on the fused dataset.

3. Methods

This section details the datasets, preprocessing process, and architecture of the proposed lwRPPC-TCN, followed by the knowledge distillation methodology.

3.1 Datasets

Two public fall datasets are used to validate the performance of the fall prediction model. They are SisFall [9] and KFall [10], as described in the following.

The SisFall [9] dataset comprises Inertial Measurement Unit (IMU) data collected from a waist-mounted sensor device containing two 3-axis accelerometers and a 3-axis gyroscope. The dataset includes 4505 samples: 2,707 samples of 19 distinct ADL types (such as walking slowly or quickly, jogging slowly or quickly, and sitting in half or low height chair, etc.) and 1,798 samples of 15 distinct fall types (such as forward falls, backward falls, and lateral falls, etc.). The sampling frequency is 200 Hz. Utilizing the data from one of the acceleration sensors (±16G) and the gyroscope(±2,000°/s).

The KFall dataset [10] collects data via a wearable waist-mounted inertial sensor device, which consists of a 3-axis accelerometer, a 3-axis gyroscope, and a 3-axis magnetometer. The dataset comprises 5,075 samples, 2,729 samples of 21 distinct ADL types (most types of the ADLs are same as SisFall, but there exist minor differences, such as standing for 30 s, picking up an object from the floor, and lying on the bed for 30 s, etc.), and 2,346 samples of 15 distinct fall types (the fall types are the same as the SisFall). The sampling frequency is 100 Hz.

3.2 Data Preprocessing

The inherent scarcity of fall samples within individual datasets will lead to insufficient training instances and limited scenario diversity. To address this limitation, a new dataset has been adopted that fuses KFall and SisFall. During data fusion, the sampling frequencies, sensor specifications, and data formats of the two datasets are first aligned to ensure that the data from different sources are comparable in terms of time domain and magnitude. Although SisFall and KFall employ sensors with differing sampling frequencies, their sensor measurement ranges, three-dimensional orientations, and the wearing positions of the acquisition devices are entirely identical. To reconcile this discrepancy, the SisFall dataset is downsampled using a quadratic linear interpolation method, followed by range normalization to align the datasets, where all data points are divided by the maximum value within the operational range of the sensor. The normalization equation is as follows:

$$X_{\text{norm}} = X_{\text{raw}} / \text{Range}_{\text{max}}. \tag{1}$$

In (1), X_{raw} represents the raw data extracted from the above two datasets, X_{norm} is the data obtained after normalization, and the maximum measurement range of the sensors is represented by $\text{Range}_{\text{max}}$. After normalization, X_{norm} will be converted into a format suitable for input to the lwRPPC-TCN.

Considering the computational limitations of wearable devices, a sliding data window is employed as the input of the model. The fall impact labels of the fall samples, the lead time for intercepting the fall data, and the length of the window size are three key factors that will influence the performance of the model. Given that a safe airbag needs about 200 ms to operate [25], and model inference time, selecting 250 ms for our model's fall prediction lead time. Since the SisFall dataset lacks explicit fall impact labels, analyzing the KFall dataset first and find that the fall impact moment of most fall events coincides with the peak of the sum magnitude vector (SMV) of the acceleration data, and the impact moment of a few fall events occurs after this peak. Therefore, the same peak-based criterion is used to label fall impact time in SisFall. Considering the resource limitations of wearable devices and the performance of the model, the experimental results indicate that a window size of 750 ms is optimal. Figure 1 shows an interception of a fall window.

3.3 Model Architecture

The inverted bottleneck block in MobileNet V2 [3] consists of two Pointwise (PW) convolutions and one Depthwise (DW) convolution. Through efficient channel expansion and contraction design, along with the application of Depthwise Separable convolution, the block significantly enhances computational efficiency without sacrificing performance, making the model fit for lightweight neural network design. However, for wearable devices with extremely limited memory, the large-channel DW convolution in the middle layer of the inverted bottleneck block still exceeds the memory capacity. In order to address

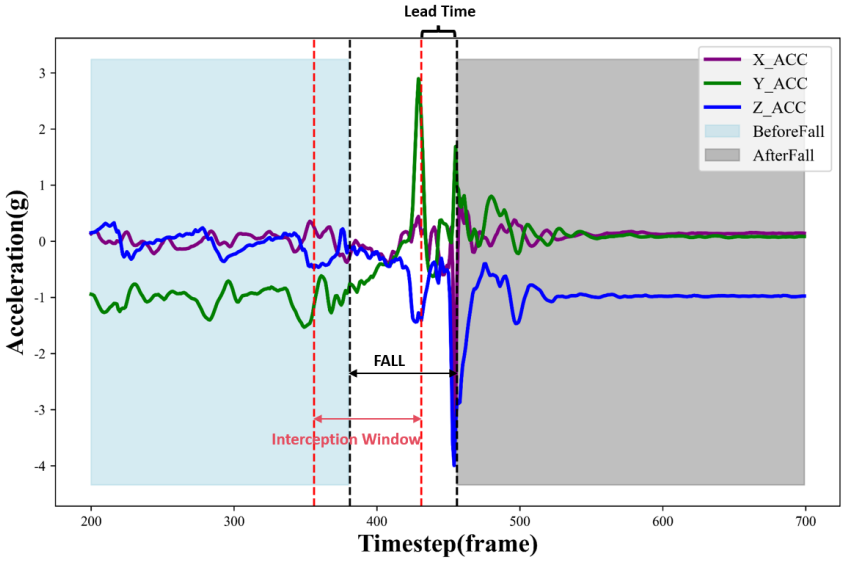

Fig. 1. Example of Interception Window (KFall). The blue area represents the ADL data before the fall, the gray area represents the data after the fall, the data during the fall is represented between the black dashed lines, and the data in the intercepted window is represented between the red dashed lines.

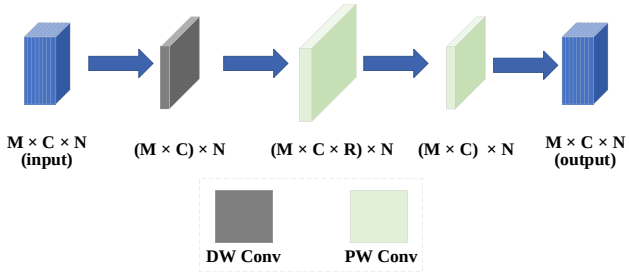

Fig. 2. ConvNeXt block, DW Conv and PW Conv represent Depthwise Convolution and Pointwise Convolution, M, C, N, and R denote the number of variables, the number of channels after embedding, the length of the timeseries data after embedding, and the expansion ratio of the PW Conv.

this problem, ConvNeXt V1 [4] proposes the ConvNeXt block (Fig. 2), which reduces the number of parameters and inference latency by forwarding the DW convolution, making it more suitable for resource-constrained mobile devices.

Learned from ConvNeXt, an lwRPPC block is proposed (Fig. 3), which extends the ConvNeXt block by incorporating structural reparameterization [26] techniques. A combined structure of large and small kernel convolutions at the DW convolutional layer is introduced to capture the long-term and short-term feature dependencies in the time-series data better. Additionally, two parallel Group PW convolutional branches [27] replace the serial PW convolutional layer in the ConvNeXt block. Ultimately, the outputs of the two parallel convolutional branches are summed and added to the input features via a residual connection to generate the output at last stage. The specific design details of this block will be elaborated in the following sections.

The overall fall prediction model architecture (Fig. 4) consists of an embedding layer, two lwRPPC block layers, a downsampling layer, a de-embedding layer, 4 TCN block layers, and a classification layer.

Embedding Layer: The extraction of spatial features from time-series data is crucial for improving the performance of the model. However, due to discrepancies in multi-source sensor data, such as accelerometers and gyroscopes, mixing multivariate variables directly will lead to confusion with irrelevant information, resulting in the loss of univariate independence. Each variable is embedded separately to enhance univariate independence and expressiveness while also capturing cross-variable dependencies. Specifically, the input time-series data $X_{\text{in}} \in \mathrm{R}^{M \times L}$, first undergoes an unsqueeze operation to produce $X_{\text{unsqueeze}} \in \mathrm{R}^{M \times 1 \times L}$. It then passes through the embedding layer to generate $X_{\text{emb}} \in \mathrm{R}^{M \times C \times N}$, where C is the embedded dimensions, and N is the embedded sequence length. This embedding process preserves the variable dimensions while enhancing the representation of univariate variables, which facilitates the subsequent extraction of cross-variable feature dependencies and learning of univariate expressions.

Lightweight Re-parameters-Parallel Convolutional Block: Following the lightweight design principle of the ConvNeXt block (Fig. 2), which moves the DW convolution in the inverted bottleneck block forward before the first PW convolution, an lwRPPC block is proposed. This block is more suitable for feature extraction and processing of time-series data. First, the 1D convolution is used instead of the 2D convolution, since time-series data is inherently one-dimensional. 1D convolution slides the filter only along the time axis, directly capturing local dependencies between neighboring time steps in the sequence without introducing spatial dimensions unrelated to the temporal information. Furthermore, the filter size in 1D convolution is typically smaller, resulting in fewer parameters and lower

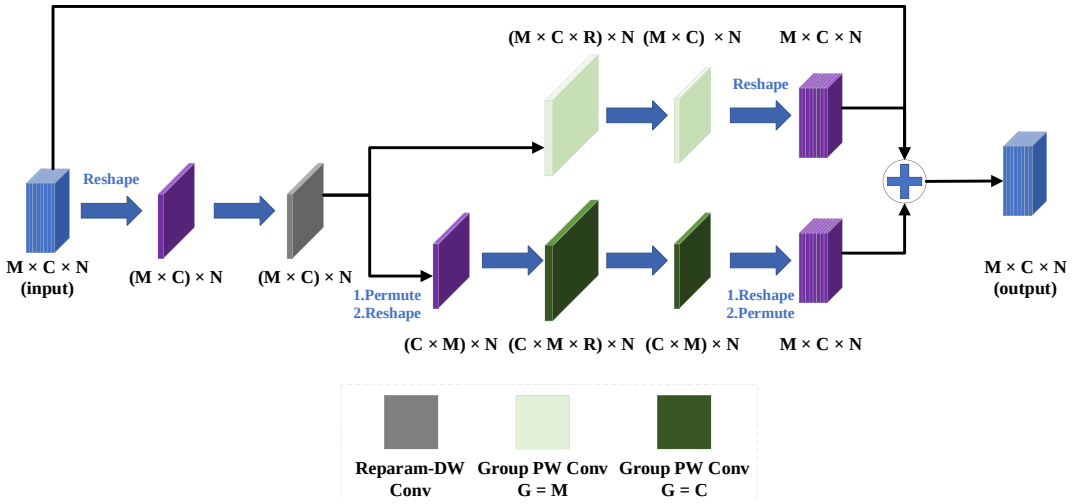

Fig. 3. lwRPPC block, Reparameter-DW Conv represents DW convolution using structural reparameterization, Group PW Conv represents PW convolution with group design, G denotes the number of convolutional groups.

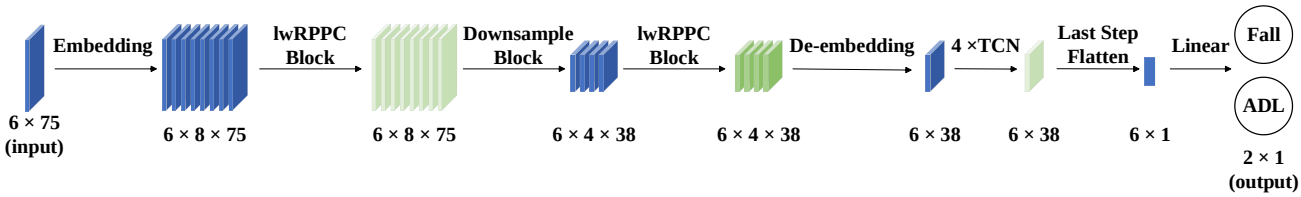

Fig. 4. lwRPPC-TCN Architecture diagram.

computational effort. This enhances computational efficiency and reduces the risk of model overfitting.

Considering the long sequence length of the temporal data, it will be challenging to capture the long-term dependency when extracting features by simply using a small convolution kernel. Such a problem will be effectively addressed through structural reparameterization techniques [26] (which combine large convolution kernels [28] and small convolution kernels). The large kernel convolution is adept at capturing long-term feature dependencies in the temporal data, while the small kernel convolution excels at extracting local feature dependencies. The structural reparameterization technique leverages the advantages of both large and small kernel convolutions, enabling the model to capture both long-term and short-term dependencies in the temporal data.

In the ConvNeXt block, after the first DW convolution layer completes feature fusion, the result was directly fed into the single-branch serial PW convolution to learn univariate expressions and multivariate dependencies jointly. That approach was prone to poor performance. In contrast, the lwRPPC block (Fig. 3) uses a multi-branch structure while replacing the PW convolution with Group PW convolution. These two Group PW convolutions with dynamic tensor reshaping are used to learn univariate expression forms and multivariate dependencies, respectively, by varying the input shape and the number of groups. Specifically, the serial Group PW convolution on the top side of Fig. 3 changes the input shape to $(M \times C) \times N$ and utilizes the Group PW convolution with Group as M to learn the univariate expression form. In contrast, the serial Group PW convolution on the bottle side changes the input shape to $(C \times M) \times N$, using the Group PW convolution with the number of groups set to C to learn the multivariate interdependencies. Learning spatial features through this multi-branch Group PW convolution complements the learning of temporal features by Reparameter-DW. Both Reparameter-DW and the branching serial Group PW convolution only extract features in one dimension of temporal, univariate, and multivariate, decoupling the extraction of temporal and spatial features from the time-series data. Eventually, a residual connection is introduced to enhance the stability and overall performance of model training.

Downsample Block: The Downsample block is a standard 1D convolutional layer, which is designed to reduce the time-series length and the number of channels, effectively decreasing the model size and inference latency.

De-embedding Layer: The de-embedding layer serves as the inverse of the channel expansion performed by the embedding layer. Using a learned weight matrix, it projects the multichannel representations of univariate inputs back to a space that corresponds one-to-one with the original sensor axes. This transformation preserves the feature information extracted in the previous layers while enabling the subsequent TCNs to perform convolution and causal modeling more directly on the original time-series frames.

TCNs: Although structural reparameterization expands the receptive field (RF) of DW convolution, enabling

the extraction of both long-term and short-term temporal features, a large number of layers are still required to stack and achieve a global model in the time dimension. TCNs, on the other hand, achieve exponentially larger receptive fields with fewer layers by stacking dilated convolutions. Equation (2) and Equation (3) are the equations for calculating the RF of DW convolution and TCN, respectively, where *layer* represents the number of stacked layers and *k* is the kernel size. TCNs are introduced to address the limitations of DW convolution in modeling global temporal dependencies, thereby enhancing the extraction of temporal features. While DW convolution excels at capturing instantaneous local features from multivariate sensor signals, TCNs perform global modeling in the time dimension. The combination of both techniques will allow the model to recognize both rapid changes over short periods and subtle trends over longer periods, improving the accuracy of fall predictions. Since TCNs converge features to the last step, the last step output of the last layer of TCNs is used as input to the classification layer.

$$\mathrm{RF}_{DW\_layer} = \mathrm{RF}_{DW\_layer-1} + (k-1), \tag{2}$$

$$\mathrm{RF}_{\mathrm{TCN}} = 1 + (k-1)(2^{layer} - 1). \tag{3}$$

Knowledge Distillation: Knowledge Distillation (KD) [8] transfers the knowledge from a teacher model to a student model effectively, enabling the student model to learn more implicit features while maintaining a small model size. This improves the performance of the student model, achieving a balance between performance and the number of parameters. During the training process, the loss function of KD enables the student model to learn from the teacher model. The $\mathrm{Loss}_{\mathrm{KD}}$ [22] is shown in (4) and consists of two parts: $\mathrm{Loss}_{\mathrm{hard}}$ and $\mathrm{Loss}_{\mathrm{soft}}$ [22]. α is a hyperparameter that controls the trade-off between hard and soft losses; as α increases, the student model places more emphasis on learning from the teacher model.

$\mathrm{Loss}_{\mathrm{hard}}$ is the cross-entropy loss function (5), where the student model learns the distribution of real labels. y represents the probability of each category in the true label and $\hat{y}$ represents the probability of each category predicted by the student model. $\mathrm{Loss}_{\mathrm{soft}}$ is a loss function based on Kullback-Leibler (KL) divergence, defined in (6). The student model learns from the teacher model through $\mathrm{Loss}_{\mathrm{soft}}$, where b, P_{t}, P_{s} denote the batch size, the prediction distribution from the teacher model, and the prediction distribution from the student model, respectively. T is a temperature hyperparameter to regulate the prediction distribution of the teacher model; the larger T is, the smoother the prediction distribution becomes, allowing the student model to gain more information beyond just the real label.

$$\mathrm{Loss}_{\mathrm{KD}} = \alpha \times \mathrm{Loss}_{\mathrm{hard}} + (1-\alpha) \times \mathrm{Loss}_{\mathrm{soft}}, \tag{4}$$

$$\mathrm{Loss}_{\mathrm{hard}} = \sum_{i=1}^{n} - y_i \log \hat{y}_i, \tag{5}$$

$$\mathrm{Loss}_{\mathrm{soft}} = T^2 \times \frac{1}{b} \times \sum_{i=1}^{b} \frac{P_t}{T} \times \log \frac{(P_s / T)}{(P_t / T)}. \tag{6}$$

Four experiments are conducted, as described below: (1) Compare the effects of different window sizes on model performance using the fused dataset to determine the optimal window size. (2) Evaluate the performance of the proposed lwRPPC-TCN on SisFall, KFall, and the fused dataset using a five-fold cross-validation method. (3) Design a large model lwRPPC-TCN-L and a small model lwRPPC-TCN-S and validate the effectiveness of the homogeneous model knowledge distillation method. (4) Compare the lwRPPC-TCN with conventional lightweight networks (lightweight to meet the requirements of wearable device deployments) and other state-of-the-art fall prediction models to comprehensively evaluate the strengths and weaknesses of each model in the fall prediction task.

3.4 Experimental Environment and Parameter Settings

The model architecture is implemented using Pytorch version 2.3.0 and Python 3.9.19. Training and testing were conducted on an Intel i5-13500H processor, with a hardware configuration of 16 GB of RAM and an NVIDIA RTX 4060 GPU. To test the performance of the proposed models, the experimental parameters are set to the base settings of the optimizer, choosing the AdamW algorithm, the learning rate is set to 0.0001, and the cross-entropy loss function is chosen. The batch size and the number of training rounds are 128 and 300, respectively. The T and the α of $\mathrm{Loss}_{\mathrm{KD}}$ are set to 10 and 0.5, respectively.

3.5 Evaluation Metrics

To objectively evaluate the performance of the proposed algorithm, several metrics for generalized model evaluation are used, including Accuracy (Acc), Sensitivity (Sen), Specificity (Spe), Precision (Prec), and F1 Score (F1). The equations [16] for these metrics are defined as follows:

$$\mathrm{Acc} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}, \tag{7}$$

$$\mathrm{Sen} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \tag{8}$$

$$\mathrm{Spe} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}, \tag{9}$$

$$\mathrm{Prec} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}, \tag{10}$$

$$\mathrm{F1} = 2 \times \frac{\mathrm{Prec} \times \mathrm{Sen}}{\mathrm{Prec} + \mathrm{Sen}}. \tag{11}$$

4. Results Analysis and Discussion

In order to assess the generalization ability of the model and reduce the risk of overfitting due to accidental

dataset partitioning, a five-fold cross-validation method is used, where the dataset is randomly split into five subsets, with 80% of the data used for training and 20% for testing in each fold. Due to the influence of fall prediction lead time and window size, the final number of intercepted data contains a total of 4,138 fall data samples (1,797 from SisFall and 2,341 from KFall) and 5,431 ADL data samples (2,702 from SisFall and 2,729 from KFall).

4.1 Evaluation Results of Window Size

To investigate the impact of window size on model performance, different window sizes are selected for comparison experiments. Given that the entire falling action lasts only about 0.8 s, the window size between 0.5 s and 1 s can capture the key information about the human posture transformation before a fall event happens. As shown in Fig. 5, the model performs optimally with a window size of 0.75 s, achieving the following metrics: accuracy of 98.57%, sensitivity of 98.60%, specificity of 98.54%, precision of 98.11%, and F1 Score of 98.35%. The figure clearly illustrates that as the window size increases or decreases; the model performance decreases correspondingly. Therefore, in subsequent experiments, the window size is set to 0.75 s.

4.2 Comparison of Models Used in Fall Prediction

To further validate the performance of the proposed lwRPPC-TCN, comparing it with some other deep learning models for fall prediction. Table 1, Table 2, and Table 3 show the evaluation results of each model. These results highlight the strengths and weaknesses of each model, despite potential differences in data preprocessing methods across frameworks.

Table 1 presents the performance comparison of different models on the KFall dataset. From the comparison results, the proposed lwRPPC-TCN-S model performs the best compared to other models with 99.13% accuracy, 99.57% sensitivity, and 98.76% specificity. It is noteworthy that FDSNeXt, augmented by the multiscale and multi-branch block (MK), ranks second only to the proposed model in terms of prediction performance. The good performance of these two models further highlights the substantial advantages of the multiscale and multi-branch design in feature extraction. The 1DConv-LSTM model, which is widely used in the field of fall, performs well in terms of sensitivity with 99.32%, second only compared to the lwRPPC-TCN. Additionally, LSTM cannot process data in parallel, leading to increased inference time, which may affect the timeliness of predictions.

Overall, the accuracy of all models exceeded 98%. However, almost all models suffer from the prediction imbalance problem. Analysis of the reasons revealed that there are ADLs like fall events in the KFall dataset, such as quickly sitting down, lying down, and jogging. Using data visualization, finding that the data and the magnitude of change of these actions at certain moments are similar to the fall activities. Therefore, it is difficult for the model to effectively distinguish these ADL's actions from fall events, leading to an imbalanced prediction.

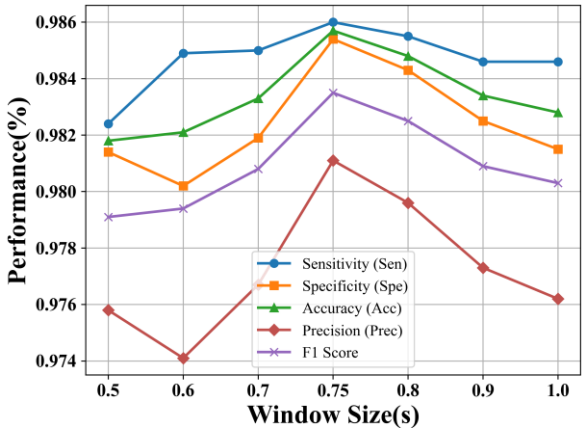

Fig. 5. Effect of window size on model performance (fused dataset).

Models	Acc (%)	Sen (%)	Spe (%)	Prec (%)	F1 (%)	Model Size (KB)
*MobileNet [3]	98.44	98.15	98.68	98.47	98.30	74
*ConvNeXt [4]	98.03	98.30	97.81	97.45	97.87	**36**
1DConv-LSTM [12]	98.00	99.32	96.84	N/A	N/A	717
PreFallKD [22]	98.05	94.79	98.53	90.62	92.66	58
*FDSNeXt [30]	98.97	99.19	**98.79**	**98.60**	98.89	283
*lwRPPC-TCN-S(ours)	**99.13**	**99.57**	98.76	98.56	**99.06**	60

Tab. 1. Comparison of the performance of different models (KFall), the models labeled * in the table are trained using the data preprocessing method, while the models not labeled * keep the predicted data reported in their original papers.

Models	Acc (%)	Sen (%)	Spe (%)	Prec (%)	F1 (%)	Model Size (KB)
*MobileNet [3]	97.11	97.30	97.01	95.57	96.42	74
*ConvNeXt [4]	96.89	96.90	96.91	95.44	96.14	**36**
spatio-temporal GNN [17]	96.10	76.20	N/A	83.50	79.70	N/A
1DConv-LSTM [29]	95.30	92.33	**98.02**	N/A	95.00	N/A
*FDSNeXt [30]	97.27	97.19	97.34	96.04	96.61	283
*lwRPPC-TCN-S(ours)	**97.62**	**97.58**	97.67	**96.54**	**97.05**	60

Tab. 2. Comparison of the performance of different models (SisFall), models labeled * in the table are trained using data preprocessing methods, while the models not labeled * keep the predicted data reported in their original papers.

Table 2 demonstrates the performance comparison of different models on the SisFall dataset. It is worth noting that the models generally perform worse on the SisFall dataset compared to the KFall dataset. However, the predictive balance of the models on SisFall is better than that on KFall. Although the spatio-temporal GNN model can extract spatio-temporal features and enhance the processing of the spatial features, its sensitivity is only 76.20%, indicating that the model struggles with predicting fall events. In contrast, the proposed lwRPPC-TCN-S achieves 97.62% accuracy, 97.58% sensitivity, 97.67% specificity, 96.54% precision, and 97.05% F1 Score on the SisFall dataset. Except for specificity, which is slightly lower than that of 1DConv-LSTM (98.02%), all other metrics exceed those of the comparison models, and the model exhibits better balance. On the other hand, the accuracy of 1DConv-LSTM is only 95.30%, and its sensitivity is only 92.33%. Both accuracy and sensitivity are significantly lower than the lwRPPC-TCN-S, indicating an imbalance in its predictions. These results indicate that the overall performance of the lwRPPC-TCN-S on the SisFall dataset is better than other compared models.

4.3 Evaluation Results of lwRPPC-TCN on the Fused Dataset and Knowledge Distillation Method

Since no previous research has conducted a fall prediction study using a dataset that incorporates both KFall and SisFall, models are reproduced from related research and apply data preprocessing methods for model training. As shown in Tab. 3, the PCNN-Transformer model achieves excellent and balanced performance on the fused dataset, with accuracy, sensitivity, and specificity all at 98.60%. Although this model slightly outperforms the lwRPPC-TCN-S in overall performance, demonstrating the advantages of the Transformer architecture and multi-scale convolution in feature extraction, it is still not as good as the lwRPPC-TCN-S with Knowledge Distillation (lwRPPC-TCN-S (KD)). Additionally, the model size of the PCNN-Transformer prevents it from being deployed in wearable devices. Finally, the large-size model called lwRPPC-TCN-L delivers the best overall performance, with an accuracy of 99.20%, a sensitivity of 99.18%, and a specificity of 99.21%.

Confusion matrices are displayed in Fig. 6, the model predicts a total of 137 erroneous samples in the tests of five-fold cross-validation, including 79 ADL samples and 58 fall samples. The performance of the lwRPPC-TCN with different parameter scales on the fused datasets is evaluated for the fall prediction task, with the experimental results shown in Fig. 7.

As shown in Tab. 1, Tab. 2, and Tab. 3, the accuracy of all models on the fused dataset is lower than that on the KFall dataset. This discrepancy primarily stems from the fact that the fusion dataset contains both fall and ADL samples from different experimental settings, resulting in a certain degree of data distribution difference between the various data sources, which leads to a decrease in performance compared to a single dataset. Nevertheless, compared with traditional lightweight models (MobileNet and ConvNeXt) and existing fall prediction models, the proposed lwRPPC-TCN-S achieves better performance on KFall, SisFall, and the fused dataset, which validates its robust generalization ability.

Overall, the introduction of the fused dataset enhances the diversity of the training data and strengthens the cross-scene adaptability of the model, albeit at the expense of some optimal performance on a single dataset.

To further validate the effectiveness of the Knowledge Distillation strategy, a large model, lwRPPC-TCN-L, is constructed by increasing the number of convolutional channels, the size of the convolutional kernel, and the number of stacked layers in the model, with hyperparameters as shown in Tab. 4. Among them, lwRPPC Layer refers to the number of stacked layers of lwRPPC block; Large Kernel size and Small Kernel size are arrays of the large and small convolutional kernel sizes in different block, respectively; Channel size is an array indicating the number of output feature channels for each block; expansion ratio

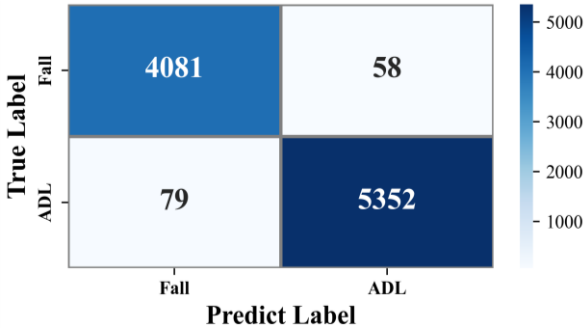

Fig. 6. Confusion matrix of lwRPPC-TCN-S (fused dataset).

Models	Acc (%)	Sen (%)	Spe (%)	Prec (%)	F1 (%)	Model Size (KB)
*MobileNet [3]	97.82	97.51	98.04	97.45	97.48	74
*ConvNeXt [4]	97.05	96.70	97.34	96.50	96.59	**36**
*PCNN-Transformer [16]	98.60	98.60	98.60	98.17	98.38	1356
*1DConv-LSTM [29]	97.87	96.33	99.04	98.71	97.50	248
*FDSNeXt [30]	97.80	97.97	97.66	96.96	97.46	283
*lwRPPC-TCN-S(ours)	98.57	98.60	98.54	98.11	98.35	60
*lwRPPC-TCN-S(KD)(ours)	98.88	98.84	98.91	98.58	98.71	60
*lwRPPC-TCN-L(ours)	**99.20**	**99.18**	**99.21**	**98.96**	**99.07**	325

Tab. 3. Comparison of the performance of different models (fused data), the models labeled * in the table are trained using the data preprocessing methods, while the models not labeled * keep the predicted data reported in their original papers.

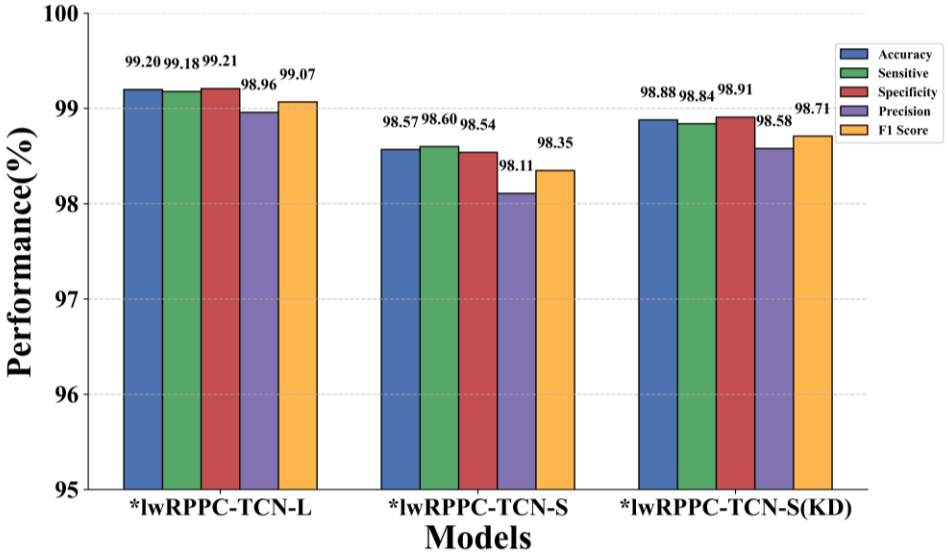

Fig. 7. Effect of knowledge distillation on model performance (fused dataset).

refers to the expansion ratio in the serial Group PW convolutional; Embedding Kernel size/stride represents the convolutional kernel size and stride of the embedding layer; Down Sample Kernel size/stride represents the convolutional kernel size and stride of the downsampling layer; De-Embedding Kernel size/stride represents the convolutional kernel size and stride of the de-embedding layer; TCN Layer represents the number of stacked layers of TCN; TCN Kernel size/stride represents the convolutional kernel size and stride of TCN; TCN Channel size is an array indicating the number of output feature channels for each TCN block; TCN Dilation represents the dilation of the TCN.

Since the structural reparameterization only merges the convolution kernels during the inference stage, lwRPPC-TCN-L is chosen not to merge the convolutional kernels as the teacher model, as its architecture is closer to the student model lwRPPC-TCN-S in the training stage, which facilitates better knowledge transfer. As shown in Fig. 7, lwRPPC-TCN-S(KD) outperforms the original lwRPPC-TCN-S in overall performance metrics: the omission rate decreases from 1.40% to 1.16%, an improvement of 17.14%, and the false alarm rate decreases from 1.46% to 1.09%, an improvement of 25.34%, which verifies that

Hyperparameter	lwRPPC-TCN-L	lwRPPC-TCN-S
lwRPPC Layer	3	2
Large Kernel size/stride	[19, 11, 7] / [1, 1, 1]	[11, 7] / [1, 1]
Small Kernel size/stride	[3, 3, 3] / [1, 1, 1]	[3, 3] / [1, 1]
Channel size	[32, 16, 8]	[8, 4]
Expansion ratio	3	2
Embedding Kernel size/stride	2 / 1	
Down Sample Kernel size/stride	2 / 2	
De-Embedding Kernel size/stride	1 / 1	
TCN Layer	4	
TCN Kernel size/stride	[7, 7, 7, 7] / [1, 1, 1, 1]	
TCN Channel size	[6, 6, 6, 6]	
TCN Dilation	2	

Tab. 4. Hyperparameter settings for lwRPPC-TCN-L and lwRPPC-TCN-S.

Models	Acc (%)	Sen (%)	Spe (%)	Prec (%)	F1 (%)
ConvNeXt-Like	97.83	98.27	97.50	96.76	97.50
ConvNeXt-Like + Re-Parameters	98.21	98.47	97.99	97.40	97.94
ConvNeXt-Like + parallel	98.19	98.24	98.16	97.60	97.92
ConvNeXt-Like + Re-Parameters + parallel	98.36	**98.62**	98.16	97.61	98.11
ConvNeXt-Like + Re-Parameters + parallel + TCN	**98.57**	98.60	**98.54**	**98.11**	**98.35**

Tab. 5. Ablation experiments on the lwRPPC block and the TCNs block (fused dataset).

the homogeneous knowledge distillation technique effectively improves the performance while controlling the model size, making it suitable for resource-constrained wearable device deployment.

5. Ablation Study

To validate the effectiveness of the proposed lwRPPC block and TCNs block, an ablation experiment is designed and compared with different blocks to better understand its impact on the performance.

To validate the effectiveness of the proposed lwRPPC block and TCNs in the fall prediction task, a series of ablation models are constructed by incrementally incorporating the structural reparameterization technique, parallel convolutional structures, and the TCNs block into the ConvNeXt-Like block. The ConvNeXt-like model incorporates the concept of decoupling within the ConvNeXt architecture. Specifically, it replaces the reparameter-DW convolution in Fig. 3 with the standard DW convolution and transforms the parallel structure into a serial structure. The experimental results are shown in Tab. 5, where the introduction of either structural reparameterization or parallel structure alone improves all model metrics compared to the ConvNeXt-Like baseline. When both techniques are ap-

plied simultaneously, performance increases further, sensitivity peaks at 98.62%, indicating that the lwRPPC block offers significant advantages in feature extraction and decoupling of cross-variate dependencies in time-series data. Additionally, the structural reparameterization technique amplifies the receptive field and improves the learning ability of long-term and short-term features. Introducing the TCN block augments global temporal modeling: although the sensitivity of the lwRPPC-TCN-S model is slightly lower than the model without TCN, it achieves the best overall results: 98.57% accuracy, 98.60% sensitivity, and 98.54% specificity, demonstrating an optimal balance between model size and performance.

6. Conclusions

A lightweight fall prediction model is proposed, combining the advantages of ConvNeXt, GroupNet, and TCN. By adjusting the model architecture, the decoupled extraction of temporal and spatial features from time-series data is achieved, making it more suitable for processing time-series data. Table 1 and Table 2 show the comparison results of the model on the KFall and SisFall datasets, respectively. On the KFall dataset, the model achieves 99.13% accuracy. On the SisFall dataset, the accuracy is 97.62%, outperforming all the compared models. Compared with ConvNeXt, the accuracy of the model is improved by 1.10% and 0.73%, respectively. Additionally, to validate the generalization ability of the model and to address the issue of limited fall data, experiments are conducted by fusing the SisFall and KFall datasets. The knowledge distillation technique is also employed to improve the performance of the model without increasing the size of the model. The experimental results in Tab. 3 show that by fusing the two public datasets, the accuracy of the model trained without the knowledge distillation technique reaches 98.57%. After using the knowledge distillation technique, the accuracy of the model is improved to 98.88%. Moreover, the size of the proposed model is only 60 KB, which meets the requirements for deployment of wearable devices, and the fall prediction lead time is 250 ms, allowing sufficient time for model inference and activation of protective equipment, such as airbags need some time to operate. These results indicate that the lwRPPC-TCN model has significant potential in IoT and smart healthcare applications. Future research will explore ways to minimize inference energy consumption, ultimately enabling the efficient deployment of IoT-based wearable devices.

Acknowledgments

This work is partly supported by the Research Fund of Education Department of Zhejiang Province, China (Professional master's degree No. Y202456356). All authors would like to thank the professors, editor, and reviewers for their support.

Data Availability

The data used to support the findings of this study are available in [9], [10].

The data and code that support the findings of this study are available from the first author, upon reasonable request.

References

[1] WORLD HEALTH ORGANIZATION (WHO). *Fragility Fractures.* [Online] Cited 2025-12-02. Available at: https://www.who.int/news-room/fact-sheets/detail/fragility-fractures.

[2] CHACCOUR, K., DARAZI, R., EL HASSANI, A. H., et al. From fall detection to fall prevention: A generic classification of fall-related systems. *IEEE Sensors Journal*, 2017, vol. 17, no. 3, p. 812–822. DOI: 10.1109/JSEN.2016.2628099

[3] SANDLER, M., HOWARD, A., ZHU, M., et al. Mobilenetv2: Inverted residuals and linear bottlenecks. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City (UT, USA), 2018, p. 4510–4520. DOI: 10.1109/CVPR.2018.00474

[4] LIU, Z., MAO, H., WU, C.-Y., et al. A ConvNet for the 2020s. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. New Orleans (LA, USA), 2022, p. 11966–11976. DOI: 10.1109/CVPR52688.2022.01167

[5] LUO, D., WANG, X. ModernTCN: A modern pure convolution structure for general time series analysis. *The Twelfth International Conference on Learning Representations (ICLR)*. Vienna (Austria), 2024, p. 31728–31770.

[6] IOANNOU, Y., ROBERTSON, D., CIPOLLA, R., et al. Deep roots: Improving CNN efficiency with hierarchical filter groups. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Honolulu (HI, USA), 2017, p. 5977–5986. DOI: 10.1109/CVPR.2017.633

[7] BAI, S., KOLTER, J. Z., KOLTUN. V. *An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling*. [Online] Cited 2025-12-03. Available at: https://arxiv.org/abs/1803.01271

[8] HINTON, G., VINYALS, O., DEAN, J. *Distilling the Knowledge in a Neural Network.* [Online] Cited 2025-12-03. Available at: https://arxiv.org/abs/1503.02531

[9] SUCERQUIA A., LÓPEZ, J. D., VARGAS-BONILLA, J. F. SisFall: A fall and movement dataset. *Sensors*, 2017, vol. 17, no. 1, p. 1–14. DOI: 10.3390/s17010198

[10] YU, X., JANG, J., XIONG, S. A large-scale open motion dataset (KFall) and benchmark algorithms for detecting pre-impact fall of the elderly using wearable inertial sensors. *Frontiers in Aging Neuroscience*, 2021, vol. 13, p. 1–14. DOI: 10.3389/fnagi.2021.692865

[11] RASSEKH, E., SNIDARO, L. Survey on data fusion approaches for fall-detection. *Information Fusion*, 2025, vol. 114, p. 1–19. DOI: 10.1016/j.inffus.2024.102696

[12] KOO, B., YU, X., LEE, S., et al. TinyFallNet: A lightweight pre-impact fall detection model. *Sensors*, 2023, vol. 23, no. 20, p. 1–14. DOI: 10.3390/s23208459

[13] YU, X., KOO, B., JANG, J., et al. A comprehensive comparison of accuracy and practicality of different types of algorithms for pre-impact fall detection using both young and old adults.

Measurement, 2022, vol. 201, p. 1–13. DOI: 10.1016/j.measurement.2022.111785

[14] VASWANI, A., SHAZEER, N., PARMAR, N., et al. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS)*. New York (USA), 2017, p. 6000–6010. DOI: 10.5555/3295222.3295349

[15] ZHOU, J., CUI, G., HU, S., et al. Graph neural networks: A review of methods and applications. *AI Open*, 2020, vol. 1, p. 57–81. DOI: 10.1016/j.aiopen.2021.01.001

[16] AL-QANESS, M. A. A., DAHOU, A., ELAZIZ, M. A., et al. Human activity recognition and fall detection using convolutional neural network and transformer-based architecture. *Biomedical Signal Processing and Control*, 2024, vol. 95, p. 1–11. DOI: 10.1016/j.bspc.2024.106412

[17] WANG, S., LI, X., LIAO, G., et al. A spatio-temporal graph neural network for fall prediction with inertial sensors. *Knowledge-Based Systems*, 2024, vol. 293, p. 1–12. DOI: 10.1016/j.knosys.2024.111709

[18] YU, X., PARK, S., KIM, D., et al. A practical wearable fall detection system based on tiny convolutional neural networks. *Biomedical Signal Processing and Control*, 2023, vol. 86, p. 1–9. DOI: 10.1016/j.bspc.2023.105325

[19] HAN, S., POOL, J., TRAN, J., et al. Learning both weights and connections for efficient neural network. In *Proceedings of the 29th International Conference on Neural Information Processing Systems (NIPS)*. MA (USA), 2015, p. 1135–1143. DOI: 10.5555/2969239.2969366

[20] CHENG, R., ZHANG, J., DENG, J., et al. Lightweight spectrum prediction based on knowledge distillation. *Radioengineering*, 2023, vol. 32, no. 4, p. 469–478. DOI: 10.13164/re.2023.0469

[21] ZHU, X., LI, J., LIU, Y., et al. A survey on model compression for large language models. *Transactions of the Association for Computational Linguistics*, 2024, vol. 12, p. 1556–1577. DOI: 10.1162/tacl_a_00704

[22] CHI, T.-H., LIU, K.-C., HSIEH, C.-Y., et al. Prefallkd: Pre-impact fall detection via CNN-ViT knowledge distillation. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Rhodes Island (Greece), 2023, p. 1–5. DOI: 10.1109/ICASSP49357.2023.10094979

[23] HAO, Z., GUO, J., HAN, K., et al. One-for-all: Bridge the gap between heterogeneous architectures in knowledge distillation. In *Proceedings of the 37th International Conference on Neural Information Processing Systems (NIPS)*. New York (USA), 2023, p. 79570–79582. DOI: 10.5555/3666122.3669605

[24] YORIOKA, D., KANG, H., IWAMURA, K. Data augmentation for deep learning using generative adversarial networks. In *2020 IEEE 9th Global Conference on Consumer Electronics (GCCE)*. Kobe (Japan), 2020, p. 516–518. DOI: 10.1109/GCCE50665.2020.9291963

[25] JUNG, H., KOO, B., KIM, J., et al. Enhanced algorithm for the detection of preimpact fall for wearable airbags. *Sensors*, 2020, vol. 20, no. 5, p. 1–13. DOI: 10.3390/s20051277

[26] DING, X., ZHANG, X., MA, N., et al. RepVGG: Making VGG-style ConvNets great again. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Nashville (TN, USA), 2021, p. 13728–13737. DOI: 10.1109/CVPR46437.2021.01352

[27] ZHANG, J., MA, Q., CUI, X., et al. High-throughput corn ear screening method based on two-pathway convolutional neural network. *Computers and Electronics in Agriculture*, 2020, vol. 175, p. 1–10. DOI: 10.1016/j.compag.2020.105525

[28] DING, X., ZHANG, X., HAN, J., et al. Scaling up your kernels to 31×31: Revisiting large kernel design in CNNs. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. New Orleans (LA, USA), 2022, p. 11953–11965. DOI: 10.1109/CVPR52688.2022.01166

[29] HATKEPOSHTI, R. K., YADOLLAHZADEH-TABARI, M., GOLSORKHTABARIAMIRI, M. Providing an approach for early prediction of fall in human activities based on wearable sensor data and the use of deep learning algorithms. *The Computer Journal*, 2024, vol. 67, no. 2, p. 658–673. DOI: 10.1093/comjnl/bxad008

[30] HNOOHOM, N., MEKRUKSAVANICH, S., JITPATTANAKUL, A. Pre-impact and impact fall detection based on a multimodal sensor using a deep residual network. *Intelligent Automation & Soft Computing*, 2023, vol. 36, no. 3, p. 3371–3385. DOI: 10.32604/iasc.2023.036551

About the Authors …

Julong PAN was born in 1965. He received his bachelor's degree and Ph.D. from Zhejiang University, China in 1987, and 2011. Now he is a Professor at the College of Information Engineering, China Jiliang University, Hangzhou, China. His main research interests are machine learning, wireless sensor network security and mobile computing.

Hongyu WANG was born in 2001. He received the Bachelor of Engineering (2023) in Computer Science and Technology from the China Jiliang University, Hangzhou, China. He is pursuing the Master of Engineering from the China Jiliang University, Hangzhou, China.

Jiyi XU was born in 2000. He received the Bachelor of Engineering (2022) in Computer Science and Technology from the China Jiliang University, Hangzhou, China. He is pursuing the Master of Engineering from the China Jiliang University, Hangzhou, China.

Haifeng XU (corresponding author) was born in 1976. He is an Associated Professor in the School of Electronic Information, LiShui Vocational & Technical College, China. His main research interests are machine learning and Internet of Things.