

Table S1. Scale of the ElderlyPipe release v1.1.

Dataset type	Dimension	Size	Remark
Videos	$1920 \times 1080 \times 3$	249 clips	30 fps, five indoor sub-areas
Static skeleton	33×4	63,855 frames	287 continuous segments, 222.5 frames on average
Dynamic skeleton	$24 \times 33 \times 4$	28,735 windows	24-frame window, stride 2

S1 Skeleton extraction with BlazePose

Skeleton extraction is built on the MediaPipe framework¹ with BlazePose², which predicts 33 three-dimensional keypoints in real time through a two-stage detector–regressor architecture. Alternative extractors such as OpenPose localise body keypoints via part affinity fields³; BlazePose is adopted here for its real-time performance on commodity hardware.

Given an input image $I \in \mathbb{R}^{H \times W \times 3}$, the first-stage detector⁴ outputs human alignment parameters: the hip centre $\mathbf{c} \in \mathbb{R}^2$, the circumscribed circle radius $r \in \mathbb{R}$ of the body, and the torso inclination angle $\theta \in \mathbb{R}$ (the angle between the line connecting the shoulders’ midpoint and the hips’ midpoint). Using these parameters, an affine transformation is applied to the image to obtain a fixed-size aligned image patch $I' \in \mathbb{R}^{256 \times 256 \times 3}$, in which the body is approximately centred, scale-normalised and upright.

The aligned patch I' is fed into the second-stage pose estimation network, an encoder–decoder trained with a multi-task objective over heatmaps, offsets and directly regressed coordinates^{2,5}. The network predicts 33 three-dimensional keypoints together with a per-keypoint visibility score and uses only the regression branch during inference. The regressed coordinates are mapped back to the original image space via an inverse affine transformation, yielding the pixel coordinates of the human keypoints.

In indoor applications, due to factors such as furniture occlusion and camera tilt angles, the keypoints obtained by MediaPipe may suffer from high-frequency jitter and missing values, making them unsuitable as high-quality gait features. This observation motivates the two-stage quality-enhancement pipeline described in the main text, which is applied in a fixed order: hierarchical occlusion compensation first completes missing coordinates, and adaptive Kalman smoothing subsequently suppresses residual jitter.

S2 Descriptions of the walking data

The scale and class structure of the unbalanced walking skeletons are summarised in Table S1.

Per-clip skeleton files. Each clip is stored in skeletons/ as one <clip_id>.npz file containing three arrays: landmarks, the $T \times 33 \times 4$ float32 array of keypoint coordinates whose four channels are (x,y,z,visibility); frame_idx, the integer frame indices within the source video; and labels, the per-frame behaviour label (one of standing, walking, using_crutch, sitting) as a string array. A labelled segment shorter than 24 frames cannot form a window: 9 of the 287 released segments are affected, including the two single-segment clips of 14 and 17 frames. These clips are released in skeletons/ as part of the raw record but produce no windows and are therefore not part of the benchmark splits.

Window index file. windows.npz contains all standardised windows as a single $(N, 24, 33, 4)$ array; windows_index.csv lists, for each window, its window_id, source clip_id, segment identifier, start frame, and the majority behaviour label (standing, walking, using_crutch, or sitting). The labels are derived by majority vote over the 24 frames of each window. The start_frame field uses original video frame indices, consistent with the segments field of metadata.csv; the frame_idx array stored in each .npz file maps every released frame to its index within the source video.

Class distribution across the release. A single video may contain more than one behaviour, so per-class video counts sum to more than 249. Across the release, walking dominates with about 58% of the frames and windows (16,600 windows), using_crutch accounts for about one quarter (7,623 windows), and sitting and standing are the smallest categories (2,649 and 1,863 windows, respectively). Frame counts are 37,224 (walking), 16,236 (using_crutch), 6,033 (sitting) and 4,362 (standing), summing exactly to the 63,855 released frames. The 287 continuous segments average 222.5 frames each.

Table S2. Comparison with representative skeleton-based action recognition datasets.

Dataset	Subjects	Scene	Unbalanced gait	Occlusion
NTU RGB+D ⁶	40, mostly young/middle-aged	Controlled lab capture (Kinect)	No	Rare, controlled
NTU RGB+D 120 ⁷	106, mostly young/middle-aged	Controlled lab capture (Kinect)	No	Rare, controlled
CarePD ⁸	Adults with Parkinson’s disease	Multi-site clinical capture	Yes (Parkinsonian)	Frequent, varied
ElderlyPipe (Ours)	Mostly residents aged 65+; some family members or caregivers	Real nursing-home indoor scenes (activity room, activity area, corridor, lobby, dining area)	Yes (using_crutch)	Frequent (furniture, backlight, dim light)

Table S3. Scene composition of the v1.1 release.

Scene	Number of clips	Lighting	Occlusion
corridor	217	mixed	none
lobby	23	backlight	none
activity_room	4	sufficient	slight
dining_area	3	dim	heavy
activity_area	2	sufficient	slight

As shown in Table S2, the essential differences from NTU RGB+D⁶, NTU RGB+D 120⁷, and CarePD⁸ lie in the subjects and the capture conditions rather than in scale. NTU series records mostly young or middle-aged subjects performing scripted actions under controlled laboratory conditions; CarePD captures Parkinsonian gait under multi-site clinical protocols. By contrast, ElderlyPipe captures elderly care recipients during natural, unscripted daily activities in real nursing-home scenes, with an explicitly annotated using_crutch category that is absent from these resources. The scene set spans five indoor sub-areas with distinct lighting and occlusion profiles, listed in Table S3. Pose-based fall-detection benchmarks⁹ continue to rely on UR-Fall¹⁰ and UP-Fall¹¹ rather than on elderly-care skeleton data of this kind.

S3 Per-clip metadata fields

metadata.csv contains one row per clip. The fields released in v1.1 are: clip_id; location (the indoor sub-area as captured by the recording file name, e.g. 1f_corridor_1 or activity_room); scene (activity_room, activity_area, corridor, lobby, or dining_area); lighting condition (sufficient / mixed / backlight / dim); occlusion level (none / slight / heavy); crutch_type (axillary for clips of using_crutch, otherwise none; forearm crutches are not annotated in the current release); n_frames (the number of released frames); segments, a semicolon-separated list of labelled spans in the form behaviour:start_frame-end_frame in original video frame coordinates (e.g. walking:271-496;standing:506-515); and capture_date and capture_timestamp (recording date and full timestamp in ISO-style format, e.g. 2025-02-24 and 2025-02-24_18-40-31, documenting capture cadence). Subject-level attributes are deliberately withheld: the release does not include subject identifiers, age bands finer than “elderly care recipients aged 65 or above, with some accompanying family members or caregivers” as described in the main text, or sex, because per-clip verification of these attributes is not available and unverified values would be misleading. Planned extensions (see the limitations discussed in the Usage Notes of the main text) will add consistent anonymised subject identifiers linking multiple clips of the same person, finer age bands, sex, and unilateral versus bilateral crutch use, once verified against the collection records.

S4 Evaluation protocol and training configuration

S4.1 Video-stratified 5-fold cross-validation with OOF aggregation

This work adopted a video-stratified 5-fold cross-validation protocol with out-of-fold (OOF) aggregation: the 247 videos that contain at least one 24-frame window are partitioned into 5 folds such that the class distribution is preserved as much as possible, and all windows extracted from the same video are assigned to exactly one fold. The remaining two released clips (14 and 17 frames) are shorter than a single window; they are part of the raw release but excluded from the benchmark (Supplementary Note S2). Each fold in turn serves as the held-out test partition while the remaining four folds are used for training; within the training folds, 15% of the videos are further held out as a validation set for early stopping and model selection.

To alleviate class imbalance, minority classes in the training folds are oversampled by at most $5\times$, and the skeleton models are trained with Focal Loss¹² ($\gamma=2$). Predictions on the five held-out folds are concatenated into a single OOF prediction set covering every sample exactly once, from which all reported metrics are computed. This protocol guarantees zero identity leakage and yields an honest estimate of generalisation to unseen videos.

The results in the main text are reported as the mean and range (max–min) across five independent training runs (seeds 42, 49, 56, 63 and 70), each rerunning the full 5-fold OOF training with its own fold split, model initialisation and augmentation randomness. The split files released with the dataset correspond to the seed-42 run; the remaining four seeds serve only to quantify run-to-run variance and are not part of the deposit.

For the two skeleton baselines, Static GCN takes single frames as samples, so its accuracy, precision, recall and F1-score are frame-level weighted averages over the OOF frames; ST-GCN takes $24 \times 33 \times 4$ windows as samples, so its metrics are window-level weighted averages over the OOF windows. In both cases AUC is the macro average of the per-class one-vs-rest AUCs. The frame-level and window-level criteria are not directly comparable, since a window must be correctly classified as a whole whereas frame-level evaluation tolerates isolated errors.

S4.2 Hardware and software configuration

Validation experiments were conducted in the following environment: Intel Core i9-14900HX CPU, NVIDIA GeForce RTX 4060 GPU (8 GB VRAM), 32 GB RAM. Software: Windows 11, Python 3.11, PyTorch 2.5.1+cu121 (main dependencies: scikit-learn 1.7.2, pandas 2.3.3, numpy 1.26.4, matplotlib 3.10.7, seaborn 0.13.2, tqdm 4.67.1).

The key training hyperparameters of the skeleton-based gait classification models are listed in Table S4. Input consists of normalised coordinates (x,y,z) and visibility c of 33 keypoints, forming a 33×4 feature matrix per frame; consecutive frames are stacked into $24 \times 33 \times 4$ dynamic windows (stride 2) for ST-GCN, while Static GCN consumes single frames.

S5 Evaluation metrics of gait classification

In evaluating unbalanced gaits, this study primarily used F1-Score and AUC to assess performance. F1-Score primarily reflects the detection capability of the model, which is defined as follows:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}$$
$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

where TP , FP and FN denote true positive, false positive and false negative respectively. F1-Score balances the trade-off between precision and recall, making it particularly suitable for imbalanced datasets.

The AUC index is the area under the Receiver Operating Characteristic (ROC) curve, which plots the relationship between true positive rate (TPR) and false positive rate (FPR) under different classification thresholds. AUC quantifies the discriminative capability of the model, and its mathematical expression is given by:

$$\text{AUC} = \int_0^1 \text{TPR}(t) d(\text{FPR}(t)),$$

where t denotes the classification threshold. AUC is threshold-independent and measures the ability of the model to distinguish between positive and negative classes across all possible decision boundaries. A higher AUC value indicates a stronger discriminative capability.

In the gait classification task, we prioritise models that achieve high F1-Score and AUC on the using_crutch category.

Table S4. Key training hyperparameters of the skeleton-based gait classification models.

Hyperparameter	Setting
Optimizer	AdamW (weight decay 10^{-3})
Initial learning rate	2×10^{-4}
LR schedule	5-epoch warmup + cosine annealing
Batch size	32
Max epochs (patience)	50 (8)
Dropout	0.5
Loss function	Focal Loss ($\gamma = 2$)
Gradient clipping norm	1.0
Hidden channels (ST-GCN / Static GCN)	64 / 128
Spatiotemporal units (ST-GCN)	3
Input window (ST-GCN)	$24 \times 33 \times 4$, stride 2
Input frame (Static GCN)	33×4
Minority oversampling cap	$5 \times$
Random seeds (5 runs)	42, 49, 56, 63, 70

S5.1 Matching between the benchmark design and the data characteristics

The design choices of the benchmark are deliberately matched to the characteristics of the data, following the principle that the evaluation protocol should reflect how the data are collected and distributed:

- Window length vs. gait dynamics. The 24-frame window (about 0.8s at 30 fps) is matched to the segment length distribution (222.5 frames on average) and to the duration of a gait cycle: one window covers roughly one stride of elderly walking, so each window contains a complete and discriminative motion unit while still being short enough to yield 28,735 training windows from 287 segments.
- Imbalance handling vs. class distribution. The oversampling cap of $5 \times$ matches the frame-level imbalance ratio (walking:standing $\approx 8.5 : 1$), so that oversampling narrows but does not invert the natural class prior; Focal Loss with $\gamma = 2$ further counteracts the observed failure mode in which the majority class dominates the decision boundary, by down-weighting the abundant, easy walking gradients.
- Occlusion augmentation vs. scene challenges. The random skin-coloured-rectangle occlusion augmentation mirrors the acquisition challenges of the dataset (furniture occlusion, backlight, dim light), so that training-time corruptions approximate the test-time conditions rather than being arbitrary.
- Sample unit vs. model family. Static GCN is evaluated at the frame level and ST-GCN at the window level; the two granularities are not directly comparable, and the numbers of the two baselines are reported separately throughout.

S5.2 Why AUC and recall can diverge on the minority class

In gait classification, the AUC and recall indices measure different perspectives. AUC is a threshold-free ranking metric that only asks whether positive samples receive higher scores than negative ones on average, whereas recall is determined by the actual decision of the model, namely under softmax classification by the argmax over class logits. A model can therefore rank using `_crutch` samples above most negatives (high AUC) while still assigning the highest logit to walking for many of them (low recall).

The category of walking accounts for roughly six tenths of the samples (37,224 of 63,855 frames; 16,600 of 28,735 windows), so the learned softmax boundary is pushed far into the minority region. Concretely, the classifiers can assign elevated scores to the `_crutch` class, which is captured by ranking-based AUC, yet still classify these samples as walking under the argmax decision, because walking retains the highest logit for almost every ambiguous sample.

Moreover, for the using_crutch-vs-rest AUC computation, the negative pool is dominated by a large number of easily distinguishable sitting and standing samples, which lifts the average rank of the positives. High AUC on an imbalanced minority class is therefore a necessary but far from sufficient condition for practical fall detection.

S6 Quantitative statistics of the quality-enhancement pipeline

Across the 63,855 released frames (2,107,215 individual keypoint values), the two-stage quality-enhancement pipeline produces the following statistics:

- 2.29% of keypoints are restored by linear interpolation across gaps shorter than $T_{\max} = 15$ frames and carry an assigned confidence of 0.6;
- 2.82% are restored by geometric estimation from the visible parent joint and carry an assigned confidence of 0.8;
- the remaining 94.9% are directly observed detections from BlazePose.

The two restoration routes together cover 5.11% of all keypoint values; the per-clip average is approximately the same. These proportions can be re-derived by users directly from the released skeletons/ files by counting the entries whose visibility channel equals 0.6 or 0.8, respectively, so the assigned confidence value of any individual keypoint can be verified from the release without additional material.

References

1. Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, et al. Mediapipe: A framework for building perception pipelines, 2019.
2. Valentin Bazarevsky, Ivan Grishchenko, Karthik Raveendran, Tyler Zhu, Fan Zhang, and Matthias Grundmann. BlazePose: On-device real-time body pose tracking. arXiv preprint arXiv:2006.10204, 2020.
3. Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2D pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(1):172–186, 2019.
4. Valentin Bazarevsky, Ivan Grishchenko, Karthik Raveendran, Tyler Zhu, Fan Zhang, and Matthias Grundmann. BlazeFace: Sub-millisecond neural face detection on mobile GPUs, 2019.
5. Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision (ECCV)*, pages 483–499, 2016.
6. Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+d: A large scale dataset for 3d human activity analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1010–1019, 2016.
7. Jun Liu, Amir Shahroudy, Mauricio Perez, Gang Wang, Ling-Yu Duan, and Alex C. Kot. NTU RGB+D 120: A large-scale benchmark for 3D human activity understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(10):2684–2701, 2020.
8. Vida Adeli, Ivan Klabucar, Javad Rajabi, et al. CARE-PD: A multi-site anonymized clinical dataset for parkinson’s disease gait assessment. In *Advances in Neural Information Processing Systems (NeurIPS 2025), Datasets and Benchmarks Track*, volume 38. Curran Associates, Inc., 2025. arXiv:2510.04312.
9. Xiaoqun Yu, Chenfeng Wang, Wenyu Wu, and Shuping Xiong. A real-time skeleton-based fall detection algorithm based on temporal convolutional networks and transformer encoder. *Pervasive and Mobile Computing*, 107, 2025.
10. Bogdan Kwolek and Michal Kepski. Human fall detection on embedded platform using depth maps and wireless accelerometer. *Computer Methods and Programs in Biomedicine*, 117(3):489–501, 2014.
11. Lourdes Martínez-Villaseñor, Hiram Ponce, Jorge Brieva, Ernesto Moya-Albor, José Núñez-Martínez, and Carlos Peñafort-Asturiano. UP-Fall detection dataset: A multimodal approach. *Sensors*, 19(9):1988, 2019.
12. Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.