Institut Riset dan Publikasi Indonesia (IRPI)
MALCOM: Indonesian Journal of Machine Learning and Computer Science
Journal Homepage: https://journal.irpi.or.id/index.php/malcom
Vol. 6 Iss. 4 October 2026, pp: 2221-2232
ISSN(P): 2797-2313 | ISSN(E): 2775-8575

Accuracy and Computational Efficiency Analysis of Deep Learning Architectures for Pose Estimation-Based Fall Detection

Analisis Akurasi dan Efisiensi Komputasi Arsitektur Deep Learning untuk Deteksi Jatuh Berbasis Pose Estimation

Ruben Emmanuel Mahardo[1], Andreas Handojo[2*]

[1,2]Program Studi Informatika, Fakultas Teknologi Industri, Universitas Kristen Petra, Indonesia

E-Mail: [1]c14220320@john.petra.ac.id, [2]handojo@petra.ac.id

Received Jun 12th 2026; Revised Jul 18th 2026; Accepted Oct 16th 2026; Available Online Oct 30th 2026
Corresponding Author: Andreas Handojo
Copyright © 2026 by Authors, Published by Institut Riset dan Publikasi Indonesia (IRPI)

Abstract

Fall incidents among older adults cause over 684,000 deaths annually worldwide, highlighting the need for automated detection systems balancing accuracy and computational efficiency. This study evaluates three sequence-based deep learning architectures for pose estimation-based fall detection: Long Short-Term Memory (LSTM), Temporal Convolutional Network (TCN), and Spatio-Temporal Graph Convolutional Network (ST-GCN). Its novelty lies in separating pose-estimation quality from post-processing effects and identifying pose extraction as the dominant real-time latency bottleneck. Pose estimation used pretrained YOLOv11-pose on the Le2i dataset, comprising 190 videos across four indoor scenarios. A 25-frame sliding window, cross-environment evaluation, independent hyperparameter grid searches, and prediction stabilization were employed. Results show that TCN achieved the best performance, with an F1-score of 1.000 in Lecture_room and a Recall of 0.824 and an F1-score of 0.875 in Office, requiring 2.97 seconds per epoch. ST-GCN matched TCN in the Office but achieved a lower Lecture_room F1-score of 0.923 and incurred 1.6 times higher training cost. LSTM offered efficient inference at 0.276 milliseconds per sequence and 17.9 FPS webcam performance, but lower Office Recall of 0.706. Pose extraction accounted for approximately 90% of pipeline latency, averaging 27 milliseconds per frame. These findings support TCN for real-time fall detection and LSTM for resource-constrained devices.

Keywords: Deep Learning, Fall Detection, LSTM, Pose Estimation, ST-GCN, TCN

Abstrak

Kejadian jatuh pada lansia menyebabkan lebih dari 684.000 kematian setiap tahun, sehingga diperlukan sistem deteksi otomatis yang menyeimbangkan akurasi dan efisiensi komputasi. Penelitian ini mengevaluasi kinerja tiga arsitektur deep learning berbasis sekuens untuk deteksi jatuh berbasis estimasi pose, yaitu Long Short-Term Memory (LSTM), Temporal Convolutional Network (TCN), dan Spatio-Temporal Graph Convolutional Network (ST-GCN). Kebaruan penelitian terletak pada pemisahan pengaruh kualitas backbone estimasi pose, stabilisasi post-processing, serta identifikasi ekstraksi pose sebagai bottleneck latensi real-time. Estimasi pose menggunakan YOLOv11-pose pada dataset Le2i yang terdiri atas 190 video dari empat skenario ruangan. Eksperimen menerapkan sliding window 25 frame, evaluasi lintas lingkungan, tuning hyperparameter melalui grid search, serta stabilisasi prediksi. Hasil menunjukkan TCN memperoleh F1-Score 1,000 pada Lecture_room serta Recall 0,824 dan F1-Score 0,875 pada Office, dengan waktu 2,97 detik per epoch. ST-GCN menyamai TCN pada Office, tetapi menghasilkan F1-Score 0,923 pada Lecture_room dengan biaya komputasi 1,6 kali lebih tinggi. LSTM menawarkan inferensi 0,276 milidetik per sekuens dan 17,9 FPS, tetapi Recall Office lebih rendah, yaitu 0,706. Ekstraksi pose menyumbang sekitar 90% latensi pipeline. Dengan demikian, TCN direkomendasikan untuk deteksi jatuh real-time, sedangkan LSTM lebih sesuai untuk perangkat dengan sumber daya terbatas.

Kata Kunci: Deep Learning, Deteksi Jatuh, LSTM, Pose Estimation, ST-GCN, TCN

1. PENDAHULUAN

Populasi lansia global diproyeksikan mencapai 1,5 miliar jiwa pada usia 65 tahun ke atas pada tahun 2050 [1]. Jatuh pada lansia menyebabkan lebih dari 684.000 kematian setiap tahunnya di seluruh dunia, dengan dampak yang lebih besar pada kelompok usia di atas 60 tahun [2]. Selain menyebabkan kematian, insiden jatuh juga mengakibatkan cedera serius, penurunan kualitas hidup, dan beban ekonomi yang signifikan akibat

perawatan medis jangka panjang [3]. Statistik ini menegaskan kebutuhan kritis akan sistem deteksi jatuh otomatis yang mampu melakukan pemantauan kesehatan lansia secara berkelanjutan. Meskipun berbagai solusi deteksi jatuh telah dikomersialkan, baik berbasis wearable device maupun sensor lingkungan, tingkat adopsinya pada lingkungan hunian mandiri lansia masih terbatas, antara lain karena akurasi yang menurun pada kondisi pencahayaan bervariasi, kepatuhan pengguna memakai perangkat wearable yang rendah, dan biaya pemasangan sensor tambahan [4]. Sistem berbasis vision yang memanfaatkan infrastruktur kamera yang sudah tersedia menjadi pendekatan yang lebih praktis untuk diadopsi secara luas.

Metode deteksi manual seperti supervisi langsung atau pelaporan mandiri memiliki efektivitas terbatas pada lingkungan hunian mandiri [4]. Keterlambatan dalam deteksi jatuh memperparah tingkat cedera dan meningkatkan risiko kesehatan [3]. Di antara pendekatan teknologi yang diusulkan, sistem berbasis vision menawarkan pemantauan non-intrusif tanpa memerlukan kontak langsung dengan pengguna dan dapat diintegrasikan ke dalam infrastruktur pemantauan lingkungan [5]. Dibandingkan dengan video RGB mentah, pose estimation memungkinkan representasi skeleton keypoint yang ringkas dan lebih tahan terhadap variasi pencahayaan serta latar belakang yang kompleks [5], [6]. Pendekatan lain menggabungkan konfigurasi multi-kamera dengan classifier machine learning konvensional untuk pengenalan pola jatuh [7].

Deteksi jatuh membutuhkan pemodelan dinamika temporal dari transisi postural (perubahan cepat dari posisi tegak ke posisi horizontal) dan bukan analisis frame individu secara terpisah [6]. Analisis berbasis satu frame berisiko salah mengklasifikasikan aktivitas normal seperti duduk, membungkuk, atau berbaring sebagai jatuh [5]. Oleh karena itu, model deep learning berbasis sequence sangat penting untuk menangkap ketergantungan temporal pada data pose secara akurat. Terdapat tiga pendekatan utama yang menonjol, yaitu Long Short-Term Memory (LSTM) [8] untuk pemodelan temporal rekuren, Temporal Convolutional Network (TCN) [9] untuk dilated causal convolution yang dapat diparalelkan, dan Spatio-Temporal Graph Convolutional Network (ST-GCN) [10] untuk pemodelan skeleton berbasis graph. Ketiga arsitektur ini dipilih karena merepresentasikan tiga paradigma pemodelan temporal yang berbeda dan saling melengkapi: LSTM sebagai baseline recurrent yang telah mapan untuk data sekuensial, TCN sebagai alternatif convolutional yang dapat diparalelkan, dan ST-GCN sebagai pendekatan graph-based yang secara eksplisit memodelkan struktur spasial skeleton. Dibandingkan pendekatan deep learning lain yang tengah berkembang seperti Transformer berbasis attention atau 3D-CNN berbasis video RGB, ketiga arsitektur ini memiliki jumlah parameter yang jauh lebih kecil dan kebutuhan data latih yang lebih rendah, sehingga lebih sesuai untuk deployment real-time pada perangkat dengan sumber daya terbatas seperti yang menjadi fokus penelitian ini.

Meskipun penelitian deteksi jatuh berbasis pose estimation telah banyak dilakukan, sebagian besar studi sebelumnya berfokus pada peningkatan akurasi klasifikasi tanpa analisis komprehensif terhadap efisiensi komputasi [5], [6], [11]. Raza dkk. [12] melaporkan ST-GCN mencapai akurasi 98,44% namun tidak menganalisis biaya komputasinya. Pada praktiknya, sistem deteksi jatuh diterapkan pada perangkat pemantauan yang menjadikan inference time, kompleksitas model, dan konsumsi sumber daya sama pentingnya dengan akurasi [4]. Selain itu, perbandingan sistematis antara model temporal murni (LSTM, TCN) dan model graph spatio-temporal (ST-GCN) masih terbatas dalam literatur [5], [6].

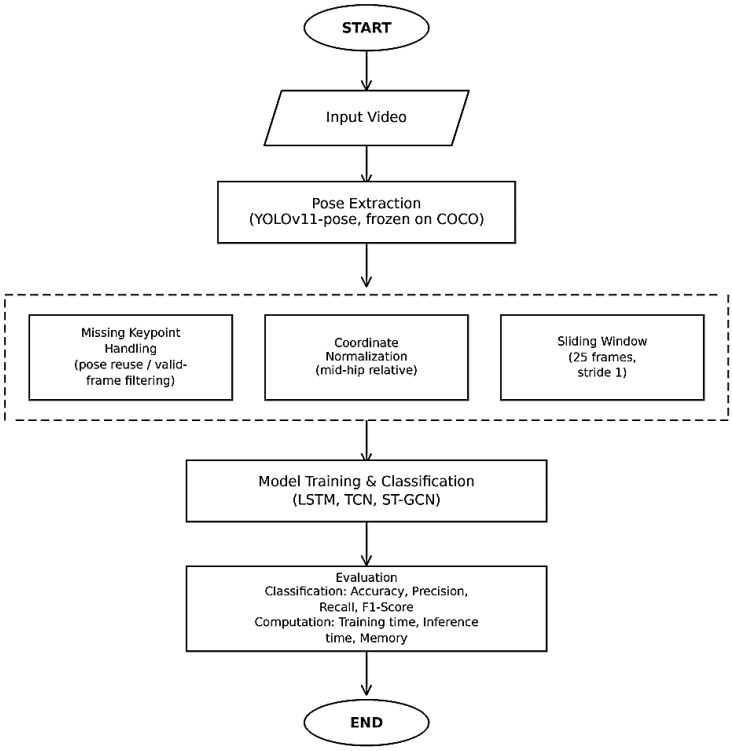
Beberapa penelitian terbaru telah mengeksplorasi variasi kombinasi pose estimation dengan arsitektur temporal untuk deteksi jatuh. Qi dan Sun [17] mengusulkan multistage framework yang mengintegrasikan TCN dengan sekuens pose 3D dan mencapai akurasi tinggi pada dataset NTU RGB+D, namun tidak mengevaluasi efisiensi komputasi maupun generalisasi cross-environment sebagaimana dilakukan pada penelitian ini. Castro dkk. [18] mengusulkan mekanisme attention pada arsitektur LSTM untuk deteksi jatuh berbasis pose estimation 2D dan 3D, menunjukkan bahwa penambahan attention meningkatkan generalisasi model dan mengurangi scene bias, namun overhead komputasi tambahan dari mekanisme attention tersebut tidak dikuantifikasi. Shin dkk. [19] mengusulkan ST-GCN tiga-stream dengan adaptive feature aggregation yang meningkatkan akurasi klasifikasi, tetapi kompleksitas arsitektur tersebut belum dibandingkan dari segi biaya komputasi terhadap pendekatan single-stream. Cho dkk. [20] mengombinasikan directed graph neural network dengan LSTM untuk deteksi jatuh antisipatif (predictive, bukan reaktif), arah berbeda dari fokus deteksi reaktif pada penelitian ini. Zobi dkk. [21] mengusulkan augmentasi data dan agregasi saat inference untuk mengatasi kegagalan generalisasi model berbasis pose estimation pada sudut pandang kamera baru dan kondisi oklusi parsial, sebuah tantangan yang relevan dengan skema evaluasi cross-environment pada penelitian ini, namun teknik agregasi saat inference tersebut menambah overhead komputasi yang tidak dianalisis dalam studi tersebut. Yan dkk. [22] membandingkan ST-GCN dengan LSTM, TCN, dan lima algoritma lain (MLP, CNN, RNN, TST, MiniRocket) untuk deteksi jatuh, namun berbasis data multiple inertial sensor (accelerometer dan gyroscope wearable), bukan pose estimation berbasis kamera, dan hanya melaporkan metrik klasifikasi (accuracy, precision, recall, F1-score) tanpa analisis efisiensi komputasi. Dibandingkan studi-studi tersebut, penelitian ini menyajikan perbandingan sistematis akurasi dan efisiensi komputasi (training time, inference time, FPS real-time) dari LSTM, TCN, dan ST-GCN berbasis pose estimation dalam satu protokol evaluasi yang identik.

Penelitian ini mengisi gap tersebut dengan melakukan analisis komparatif yang adil dan sistematis terhadap LSTM, TCN, dan ST-GCN untuk deteksi jatuh berbasis pose estimation pada dataset Le2i. Kontribusi

spesifik penelitian ini adalah: (1) evaluasi cross-environment yang secara eksplisit memisahkan kontribusi kualitas backbone pose estimation (YOLOv11n vs. YOLOv11m) dari kontribusi algoritma post-processing melalui empat tahap eksperimen terkontrol (Tabel 4), berbeda dengan studi-studi LSTM/TCN/ST-GCN yang ditinjau di atas [12], [17]-[22] yang, sepengetahuan penulis, tidak secara eksplisit mengisolasi kontribusi kualitas backbone pose estimation dari algoritma post-processing; dan (2) analisis latensi end-to-end pada pipeline real-time yang mengidentifikasi ekstraksi pose, bukan klasifikasi, sebagai bottleneck dominan (~90% dari latensi total), sebuah aspek efisiensi yang jarang dilaporkan pada studi-studi tersebut, memberikan panduan konkret untuk prioritas optimasi pada deployment nyata. Dua pertanyaan penelitian yang dijawab adalah: (1) Bagaimana performa LSTM, TCN, dan ST-GCN dalam metrik klasifikasi (accuracy, precision, recall, F1-score) dan efisiensi komputasi (training time, inference time)? (2) Apakah kompleksitas spatio-temporal pada ST-GCN memberikan peningkatan performa yang signifikan dibandingkan pendekatan temporal murni untuk deteksi jatuh real-time?

2. METODOLOGI PENELITIAN

Gambar 1 menggambarkan keseluruhan pipeline penelitian, mulai dari input video mentah melalui pose extraction, preprocessing sequence, pelatihan dan klasifikasi model, hingga evaluasi akhir. Setiap model dilatih dan dievaluasi dengan protokol identik untuk memastikan perbandingan yang adil.



Gambar 1. Pipeline Penelitian Deteksi Jatuh

2.1 Dataset

Penelitian ini menggunakan Le2i Fall Detection Dataset [13] yang terdiri dari 190 video dengan resolusi 320×240 piksel, 25 FPS, pada empat skenario ruangan dengan satu orang per video. Strategi evaluasi cross-environment diterapkan, yaitu 130 video pelatihan dari skenario coffee_room dan home, serta 60 video pengujian dari skenario lecture_room dan office yang sepenuhnya tidak dilihat selama pelatihan (Tabel 1). Konfigurasi ini mengikuti Raza dkk. [12] sehingga memungkinkan perbandingan langsung dengan penelitian sebelumnya.

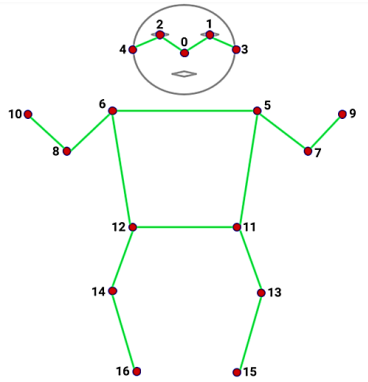
Tabel 1. Distribusi Le2i Fall Detection Dataset

Subset	Skenario	Jumlah Video
Training	Coffee_room_01	48
Training	Coffee_room_02	22
Training	Home_01	30
Training	Home_02	30
Testing	Lecture_room	27
Testing	Office	33

2.2 Ekstraksi Pose dan Preprocessing

Human pose estimation adalah tugas computer vision untuk melokalisasi titik sendi anatomis (keypoint) tubuh manusia dari citra, umumnya melalui pendekatan top-down (mendeteksi orang terlebih dahulu, kemudian mengestimasi pose pada tiap bounding box) atau bottom-up (mendeteksi seluruh keypoint pada citra kemudian mengelompokkannya per individu). Secara teoretis, representasi keypoint lebih menguntungkan dibandingkan raw pixel untuk deteksi jatuh karena mereduksi dimensi input secara drastis sehingga model klasifikasi berfokus pada dinamika kinematik tubuh, yaitu perubahan konfigurasi sendi antarwaktu yang menjadi penanda utama peristiwa jatuh, serta lebih tahan terhadap variasi penampilan subjek, pencahayaan, dan latar belakang [5], [6], [11]. YOLOv11-pose dipilih karena mengadopsi paradigma single-stage yang meregresi bounding box dan keypoint secara bersamaan dalam satu forward pass [14], menghindari overhead inferensi dua tahap; pertimbangan ini relevan karena ekstraksi pose terbukti menjadi komponen latensi dominan pada pipeline real-time penelitian ini (Bagian 3.3).

Pose estimation dilakukan menggunakan YOLOv11-pose [14] yang telah di-pretrained pada COCO [15] (frozen, tanpa fine-tuning), mengekstraksi 17 keypoint dalam koordinat (x, y) per frame mengikuti format skeleton COCO. Dua varian backbone dievaluasi, yaitu YOLOv11n-pose (nano, Training 1) dan YOLOv11m-pose (medium, Training 2). Koordinat dinormalisasi relatif terhadap posisi mid-hip sehingga representasi menjadi invarian terhadap posisi absolut subjek dalam frame. Visualisasi 17 keypoint beserta indeksnya ditampilkan pada Gambar 2 dan Tabel 2.



Gambar 2. Visualisasi 17 Keypoint Pose Tubuh Manusia Format COCO

Tabel 2. Indeks dan Nama 17 Keypoint Format COCO

Indeks	Keypoint	Indeks	Keypoint
0	nose	9	left_wrist
1	left_eye	10	right_wrist
2	right_eye	11	left_hip
3	left_ear	12	right_hip
4	right_ear	13	left_knee
5	left_shoulder	14	right_knee
6	right_shoulder	15	left_ankle
7	left_elbow	16	right_ankle
8	right_elbow		

Keypoint yang hilang ditangani melalui pose reuse untuk LSTM dan TCN (menggunakan kembali pose valid terakhir dalam window 1 detik), sedangkan ST-GCN hanya menggunakan frame dengan deteksi valid untuk menjaga relasi spasial pada graph. Sequence dibentuk menggunakan sliding window 25 frame (stride 1), menghasilkan total 36.166 window (28.045 pelatihan, 8.121 validasi). Ketidakseimbangan kelas (FALL: ~19%) ditangani dengan WeightedRandomSampler.

2.3 Arsitektur Model

Pemilihan ketiga arsitektur didasarkan pada karakteristik pemodelan temporal yang berbeda. LSTM dipilih karena mekanisme gating-nya terbukti efektif menangkap dependensi temporal jangka panjang pada berbagai tugas klasifikasi sekuens, termasuk pengenalan aktivitas manusia berbasis sensor maupun video [8], [5]. TCN dipilih karena dilated causal convolution memungkinkan receptive field temporal yang luas tanpa mengorbankan paralelisme komputasi, dan secara empiris terbukti menyamai atau mengungguli recurrent network pada berbagai benchmark pemodelan sekuens [9]. ST-GCN dipilih karena secara eksplisit memodelkan topologi skeleton sebagai graph, menangkap relasi spasial antar-sendi yang tidak direpresentasikan oleh LSTM maupun TCN yang memperlakukan keypoint sebagai vektor flat [10].

LSTM: Stacked LSTM layer dengan dropout, menerima input yang di-reshape ke bentuk (N, 25, 34) dengan 34 = 17 keypoint × 2 koordinat. Hidden state pada timestep terakhir diteruskan ke fully connected classifier [8].

TCN: Stacked blok dilated causal convolution (1D Conv) dengan BatchNorm, ReLU, dropout, dan residual connection. Faktor dilation yang meningkat secara eksponensial memperluas temporal receptive field tanpa menambah parameter, sehingga memungkinkan pemrosesan paralel yang efisien [9]. Input menggunakan format channel-first (N, 34, 25).

ST-GCN: Arsitektur berbasis graph yang merepresentasikan setiap keypoint sebagai node dan koneksi anatomis sebagai edge, sehingga struktur skeleton tetap terjaga dengan format input (N, C, T, V). Spatial graph convolution menangkap relasi antar-sendi per frame, dikombinasikan dengan temporal convolution antar frame [10]. Adjacency matrix mengikuti topologi standar skeleton COCO.

2.4 Konfigurasi Pelatihan, Post-processing, dan Evaluasi

Seluruh model dilatih dengan protokol identik menggunakan Cross-Entropy Loss dan optimizer Adam. Hyperparameter (hidden size/channels, jumlah layer untuk LSTM, dropout rate, learning rate, batch size) dituning secara independen per model melalui grid search pada pembagian train-validation 80:20 per video, dengan early stopping berbasis F1-Score validasi (patience = 10 epoch). Pelatihan akhir berjalan hingga 100 epoch dengan konfigurasi optimal hasil grid search.

Konfigurasi optimal hasil grid search untuk setiap model pada kedua training run disajikan pada Tabel 3, menunjukkan bahwa parameter yang dipilih (jumlah layer/channel yang moderat, dropout 0,2–0,3, learning rate 0,0001–0,001) konsisten dengan rentang umum yang digunakan pada literatur sequence modeling untuk data time-series berskala kecil hingga menengah [9].

Tabel 3. Konfigurasi Optimal Hasil Grid Search per Model

Run	Model	YOLO	Hidden/Channels	Layers	Dropout	LR	Batch	Epoch Stop
Training 1	LSTM	YOLOv11n	64 units	2	0,3	0,0005	64	29
Training 1	TCN	YOLOv11n	[64,64,128,128]	-	0,2	0,0005	64	14
Training 1	ST-GCN	YOLOv11n	[32,64,64]	-	0,2	0,001	64	18
Training 2	LSTM	YOLOv11m	128 units	1	0,2	0,001	32	26
Training 2	TCN	YOLOv11m	[64,64,128,128]	-	0,2	0,0005	64	14
Training 2	ST-GCN	YOLOv11m	[32,64,64]	-	0,2	0,0001	32	23

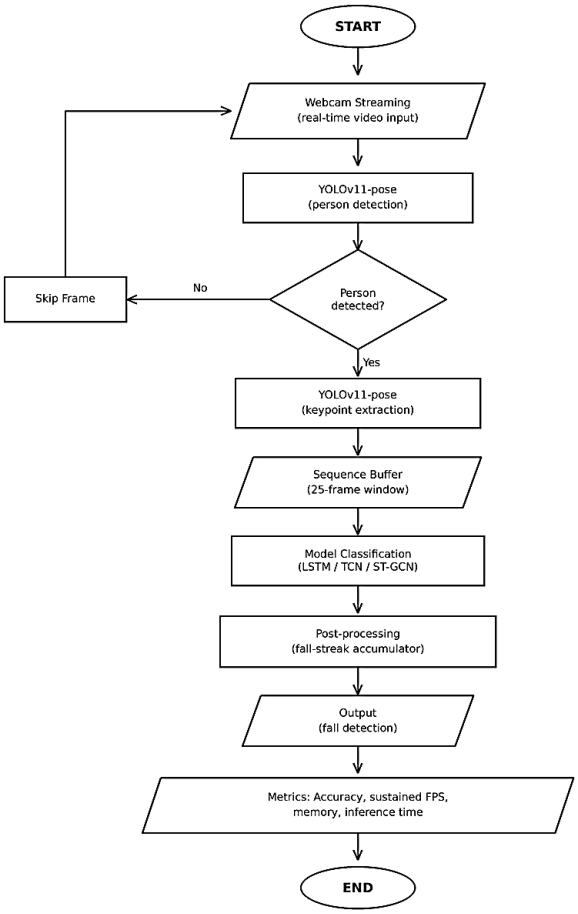
Sebuah tahap post-processing diterapkan pada prediksi per-window mentah untuk menekan false positive dari postur transien yang menyerupai jatuh. Post-processing ini menggunakan akumulator bersyarat (fall-streak): setiap prediksi positif menambah +1,0 ketika bounding box berada pada postur horizontal atau +0,3 ketika vertikal (berdiri), sedangkan setiap prediksi negatif mengurangi akumulator sebesar 1,0 dengan batas bawah 0. Sebuah FALL dipicu ketika akumulator mencapai threshold 3,0. Prosedur lengkapnya dirangkum pada Algoritma 1.

Algoritma 1. Post-processing fall-streak accumulator

```
Input: per-window prediction label, posture flag is_standing
Output: FALL alert
  Initialize fall_streak = 0, threshold = 3.0
  for each incoming frame do
    if label == FALL then
      if is_standing then
        fall_streak = fall_streak + 0.3
      else
        fall_streak = fall_streak + 1.0
      end if
    else
      fall_streak = max(0, fall_streak - 1.0)
    end if
    if fall_streak >= threshold then
      trigger FALL;  last_fall_time = now
    end if
  end for
```

Post-processing ini beroperasi di dalam loop inferensi real-time. Gambar 3 menunjukkan pipeline real-time secara lengkap: frame webcam diteruskan ke YOLOv11-pose untuk deteksi orang (frame tanpa deteksi

orang akan dilewati), keypoint diekstraksi dan dibufferkan ke sliding window 25 frame, model terpilih mengklasifikasikan setiap window, dan akumulator post-processing menentukan apakah peringatan jatuh dipicu.



Gambar 3. Pipeline Inferensi Real-time untuk Deteksi Jatuh

Performa klasifikasi dievaluasi menggunakan Accuracy, Precision, Recall, dan F1-Score dari confusion matrix pada testing set sebanyak 60 video [16]. Signifikansi statistik perbedaan akurasi antar model diuji menggunakan McNemar's test berpasangan pada prediksi per-video [16]. Recall diprioritaskan karena jatuh yang terlewat (False Negative) memiliki risiko keselamatan tertinggi [3]. Efisiensi komputasi diukur melalui training time per epoch, inference time (ms/sequence), dan FPS real-time melalui webcam streaming. Seluruh pengukuran dilakukan pada NVIDIA RTX 4060 Laptop GPU (16 GB RAM, Intel Core Ultra 7 CPU).

Untuk memisahkan kontribusi backbone pose estimation dari kontribusi tahap post-processing, evaluasi disusun ke dalam empat tahap eksperimen yang dirangkum pada Tabel 4. Training 1 dan Testing 1 menggunakan backbone YOLOv11n dengan prediksi per-window mentah (baseline). Testing 2 mempertahankan backbone yang sama tetapi menambahkan tahap post-processing untuk mengisolasi pengaruhnya. Training 2 / Testing 3 kemudian meningkatkan backbone ke YOLOv11m bersama dengan post-processing, membentuk konfigurasi final. Dalam seluruh hasil, setiap angka yang dilaporkan diberi label sesuai tahap yang menghasilkannya sehingga setiap nilai dapat dilacak ke pengaturan backbone dan post-processing spesifiknya.

Tabel 4. Konfigurasi Eksperimen

Tahap	Backbone YOLO	Post-processing	Tujuan
Training 1	YOLOv11n-pose	Tidak	Pelatihan baseline
Testing 1	YOLOv11n-pose	Tidak	Evaluasi baseline (prediksi mentah)
Testing 2	YOLOv11n-pose	Ya	Isolasi pengaruh post-processing
Training 2 / Testing 3	YOLOv11m-pose	Ya	Konfigurasi final (backbone di-upgrade)

3. HASIL DAN PEMBAHASAN

3.1 Efisiensi Pelatihan

TCN secara konsisten mencapai F1-Score validasi tertinggi pada kedua training run (0,9105 pada Training 1; 0,9124 pada Training 2; Tabel 5), diikuti LSTM (0,9093; 0,8890) dan ST-GCN (0,9039; 0,9034) yang berada tidak jauh di belakangnya. Sebagaimana didefinisikan pada Tabel 4, Training 1 menggunakan backbone YOLOv11n dan Training 2 menggunakan backbone YOLOv11m yang lebih besar. Penurunan LSTM pada Training 2 (0,9093 → 0,8890) berlawanan dengan TCN dan ST-GCN yang tetap stabil pada backbone yang sama, menunjukkan LSTM tidak konsisten merespons positif terhadap upgrade backbone.

Tabel 5. F1-Score Validasi per Training Run

Model	Training 1	Training 2
LSTM	0,9093	0,8890
TCN	0,9105	0,9124
ST-GCN	0,9039	0,9034

Tabel 6 menyajikan efisiensi pelatihan untuk Training 2 (backbone YOLOv11m). LSTM menunjukkan waktu per epoch terendah (1,73 detik) dan inference tercepat (0,276 ms/seq) karena jumlah parameternya yang kecil (84.226). ST-GCN memiliki waktu per epoch tertinggi (4,82 detik) akibat operasi graph convolution yang kurang dapat diparalelkan, meskipun jumlah parameternya (92.710) lebih sedikit daripada TCN (228.546). TCN menyeimbangkan efisiensi dan kapasitas model dengan waktu per epoch yang kompetitif (2,97 detik), didukung oleh dilated causal convolution yang dapat diparalelkan.

Tabel 6. Perbandingan Efisiensi Pelatihan (Training 2, YOLOv11m)

Model	Time/Epoch (s)	Size (MB)	Parameter	Peak VRAM (MB)	Inference (ms/seq)
LSTM	1,73	0,323	84.226	30,28	0,276
TCN	2,97	0,898	228.546	24,29	1,852
ST-GCN	4,82	0,382	92.710	20,19	1,426

3.2 Performa Klasifikasi

Tabel 7, 8, dan 9 melaporkan hasil klasifikasi per skenario untuk Testing 1, Testing 2, dan Testing 3 secara berurutan. Ketiganya merupakan tahap evaluasi yang didefinisikan pada Tabel 4. Testing 3 (YOLOv11m + post-processing) merupakan konfigurasi final dan menjadi acuan utama untuk rekomendasi deployment.

Tabel 7. Hasil Testing 1 (YOLOv11n, Tanpa Post-processing)

Dataset	Model	Accuracy	Precision	Recall	F1-Score
Lecture_room	LSTM	0,926	0,875	1,000	0,933
Office	LSTM	0,636	0,609	0,824	0,700
Lecture_room	TCN	1,000	1,000	1,000	1,000
Office	TCN	0,667	0,650	0,765	0,703
Lecture_room	ST-GCN	0,704	0,688	0,786	0,734
Office	ST-GCN	0,758	0,737	0,824	0,778

Tabel 8. Hasil Testing 2 (YOLOv11n, Dengan Post-processing)

Dataset	Model	Accuracy	Precision	Recall	F1-Score
Lecture_room	LSTM	1,000	1,000	1,000	1,000
Office	LSTM	0,727	0,700	0,824	0,757
Lecture_room	TCN	1,000	1,000	1,000	1,000
Office	TCN	0,788	0,813	0,765	0,788
Lecture_room	ST-GCN	0,778	0,786	0,786	0,786
Office	ST-GCN	0,697	0,706	0,706	0,706

Tabel 9. Hasil Testing 3 – Konfigurasi Final (YOLOv11m, Dengan Post-processing)

Dataset	Model	Accuracy	Precision	Recall	F1-Score
Lecture_room	LSTM	0,963	1,000	0,929	0,963
Office	LSTM	0,818	0,923	0,706	0,800
Lecture_room	TCN	1,000	1,000	1,000	1,000
Office	TCN	0,879	0,933	0,824	0,875
Lecture_room	ST-GCN	0,926	1,000	0,857	0,923
Office	ST-GCN	0,879	0,933	0,824	0,875

Pada konfigurasi final (Tabel 9), TCN mencapai performa sempurna pada Lecture_room (F1-Score = 1,000) dan hasil terbaik pada Office (Recall = 0,824, F1-Score = 0,875). ST-GCN menyamai TCN secara persis pada Office, dengan keempat metrik identik (Accuracy 0,879, Precision 0,933, Recall 0,824, F1-Score 0,875), tetapi tertinggal pada Lecture_room (F1-Score 0,923 vs. 1,000). LSTM menunjukkan efisiensi yang kuat namun Recall Office yang lebih rendah (0,706), berarti 29,4% kejadian jatuh terlewat pada konfigurasi final. Uji McNemar berpasangan pada prediksi per-video (n=60) menunjukkan bahwa perbedaan akurasi antar ketiga model pada konfigurasi final ini tidak signifikan secara statistik (LSTM vs. TCN: 1 vs. 4 video diskordan, p=0,375; LSTM vs. ST-GCN: 2 vs. 3 video diskordan, p=1,000; TCN vs. ST-GCN: 3 vs. 1 video diskordan, p=0,625), sehingga keunggulan TCN pada metrik akurasi bersifat indikatif, bukan konklusif. Jumlah pasangan diskordan yang kecil (4-5 video) pada ukuran sampel n=60 membatasi daya statistik uji ini untuk mendeteksi perbedaan yang lebih kecil, sehingga hasil ini tidak dapat diinterpretasikan sebagai bukti kesetaraan performa antar model.

Perbandingan antar tahap mengisolasi pengaruh setiap pilihan desain. Penambahan post-processing pada backbone YOLOv11n (Tabel 7 → Tabel 8) meningkatkan Precision secara substansial pada LSTM dan TCN (misalnya, Precision Office TCN: 0,650 → 0,813) tanpa mengorbankan Recall, dengan menekan false alarm dari postur transien yang menyerupai jatuh. Sebaliknya, tahap yang sama justru sedikit menurunkan F1-Score Office ST-GCN dari 0,778 menjadi 0,706 karena aktivasi akumulator FALL yang valid sebagian ikut tersaring. Upgrade backbone dari YOLOv11n ke YOLOv11m (Tabel 8 → Tabel 9) menghasilkan peningkatan paling besar pada ST-GCN (F1-Score Office: 0,706 → 0,875), menunjukkan ketergantungan ST-GCN pada kualitas keypoint lebih tinggi dibandingkan dengan LSTM dan TCN.

Gambar 4 menggambarkan hasil ini secara kualitatif sebagai sequence 12 panel prediksi TCN pada video jatuh Lecture_room (Testing 3). Model mempertahankan prediksi Normal selama subjek masih tegak dan pada fase awal penurunan (panel a–j), kemudian beralih ke FALL yang terkonfirmasi setelah subjek mencapai lantai dan akumulator post-processing melewati threshold-nya (panel k–l).



Gambar 4. Contoh sequence prediksi TCN pada video jatuh Lecture_room: (a) tidak ada orang terdeteksi, (b-j) postur normal, (k-l) jatuh terdeteksi

3.3 Efisiensi Komputasi

Tabel 10 melaporkan performa real-time webcam streaming. Ekstraksi YOLOv11-pose rata-rata 27,3–27,9 ms per frame, sedangkan inference klasifikasi berkisar 0,41 hingga 3,16 ms, sehingga ekstraksi pose menyumbang sekitar 90% latensi per frame. Perbedaan inference time antar classifier LSTM, TCN, dan ST-GCN dengan demikian memberikan dampak minimal terhadap FPS sistem secara keseluruhan, dan optimasi latensi sebaiknya difokuskan pada ekstraksi pose, bukan pada tahap klasifikasi.

Tabel 10. Performa Real-time Webcam Streaming

Model	FPS	Ekstraksi Pose (ms/frame)	Inference Klasifikasi (ms)	VRAM (MB)
LSTM	17,9	27,6	0,41	115
TCN	16,5	27,9	3,16	115
ST-GCN	16,2	27,3	2,28	115

LSTM mempertahankan laju streaming tertinggi (17,9 FPS), diikuti TCN (16,5 FPS) dan ST-GCN (16,2 FPS); selisih yang kecil ini mencerminkan dominasi YOLOv11m-pose pada waktu pipeline. Konsumsi VRAM identik sebesar 115 MB pada ketiga model, sepenuhnya berasal dari YOLOv11m-pose, mengindikasikan karakteristik deployment yang stabil dan berkelanjutan.

3.4 Validasi Real-time

Mengikuti pipeline inferensi real-time pada Gambar 3, validasi webcam streaming menggunakan kamera Logitech C270 (dipasang pada ketinggian ~232 cm) mengonfirmasi bahwa LSTM dan TCN mengklasifikasikan dengan benar kelima skenario uji: dua kejadian jatuh (jatuh dari kursi, jatuh ke belakang dari posisi berdiri) dan tiga aktivitas ADL (mengambil benda dari kursi, membungkuk untuk mengambil benda, jongkok). ST-GCN mendeteksi seluruh kejadian jatuh dengan benar tetapi menghasilkan false positive pada dua aktivitas ADL yang melibatkan gerakan tubuh ke bawah secara cepat, konsisten dengan sensitivitas model graph terhadap konfigurasi skeleton spasial yang sementara menyerupai postur jatuh.

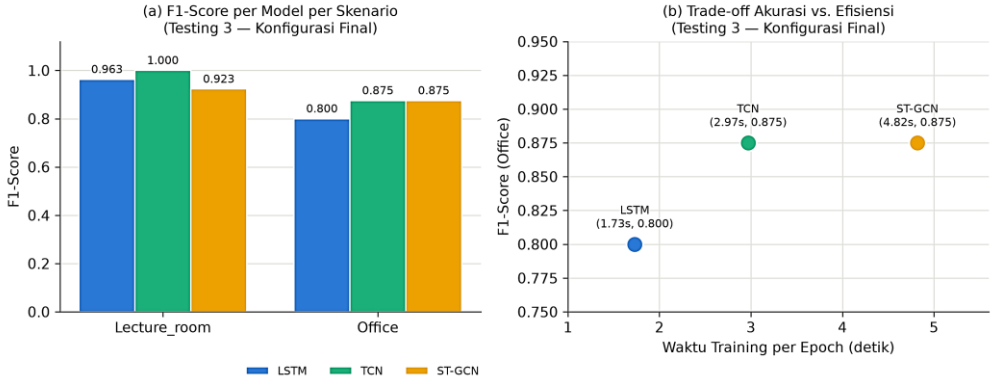
3.5 Analisis Trade-off

Tabel 11 merangkum trade-off akurasi-efisiensi secara keseluruhan pada konfigurasi final (Testing 3). Menjawab pertanyaan penelitian 1, ketiga model menunjukkan profil yang berbeda secara jelas: LSTM paling efisien tetapi paling tidak akurat pada konfigurasi final; TCN mencapai keseimbangan terbaik antara akurasi dan efisiensi; ST-GCN menyamai akurasi TCN pada Office tetapi dengan biaya komputasi tertinggi.

Tabel 11. Ringkasan Trade-off – Konfigurasi Final (Testing 3)

Model	Recall Office	F1-Score Office	Time/Epoch (s)	Inference (ms/seq)	FPS Webcam
LSTM	0,706	0,800	1,73	0,276	17,9
TCN	0,824	0,875	2,97	1,852	16,5
ST-GCN	0,824	0,875	4,82	1,426	16,2

Menjawab pertanyaan penelitian 2, kompleksitas spatiotemporal ST-GCN tidak memberikan keunggulan yang signifikan dibandingkan TCN. TCN mengungguli ST-GCN pada Lecture_room (F1-Score 1,000 vs. 0,923) dan menyamainya secara persis pada Office, tanpa pernah dikalahkan pada skenario mana pun yang dievaluasi, meskipun uji McNemar menunjukkan perbedaan ini tidak signifikan secara statistik (p=0,625, Bagian 3.2). Meskipun ST-GCN memiliki parameter 2,5× lebih sedikit daripada TCN (92.710 vs. 228.546), efisiensi parameter ini tidak diterjemahkan menjadi efisiensi komputasi: operasi graph convolution kurang dapat diparalelkan dibandingkan dilated causal convolution pada TCN, sehingga menghasilkan training time per epoch tertinggi (4,82 detik) dan FPS real-time terendah (16,2) di antara ketiga model. Hasil ini sejalan dengan karakteristik TCN yang menggunakan dilated causal convolution untuk menangkap ketergantungan temporal jangka pendek dan panjang dengan paralelisasi yang baik [9]. Gambar 5 merangkum perbandingan F1-Score dan trade-off akurasi-efisiensi ini secara visual.



Gambar 5. Perbandingan F1-Score per Model per Skenario dan Trade-off Akurasi-Efisiensi pada Konfigurasi Final (Testing 3)

3.6 Diskusi

Selain karakteristik dilated causal convolution yang telah dibahas pada Bagian 3.5, keunggulan TCN juga dapat dijelaskan oleh ketiadaan rekursi sekuensial seperti pada LSTM dan ketiadaan batasan topologi graph tetap seperti pada ST-GCN, sehingga model dapat mempelajari pola transisi postural pada berbagai skala waktu secara paralel. Pada konteks deteksi jatuh, keunggulan ini tercermin secara deskriptif pada F1-Score

sempurna (1,000) di Lecture_room dan performa terbaik pada Office (Recall 0,824), yang konsisten pada kedua backbone YOLOv11n dan YOLOv11m (Tabel 5), meskipun sebagaimana ditunjukkan uji McNemar pada Bagian 3.2, perbedaan agregat antarmodel tidak signifikan secara statistik.

Sebaliknya, kompleksitas spatio-temporal ST-GCN tidak diterjemahkan menjadi keunggulan performa yang signifikan. Terdapat dua penjelasan yang saling melengkapi. Pertama, dataset Le2i menyediakan skeleton tunggal per video tanpa interaksi multi-orang atau oklusi parsial yang kompleks, kondisi yang menjadi kekuatan utama ST-GCN pada tugas pengenalan aksi berbasis graph [10]. Tanpa variasi relasi spasial antar-sendi yang kaya, keunggulan pemodelan graph menjadi kurang relevan dibandingkan pemodelan temporal murni. Kedua, operasi graph convolution pada ST-GCN kurang dapat diparalelkan dibandingkan convolutional 1D pada TCN, menghasilkan waktu pelatihan per epoch tertinggi (4,82 detik) meskipun jumlah parameternya 2,5 kali lebih sedikit (Tabel 6), mengindikasikan bahwa efisiensi parameter tidak selalu berkorelasi dengan efisiensi komputasi aktual pada perangkat keras GPU standar.

Temuan bahwa upgrade backbone dari YOLOv11n ke YOLOv11m memberikan peningkatan performa terbesar pada ST-GCN (F1-Score Office: 0,706 → 0,875, Tabel 8 vs. Tabel 9) memperkuat interpretasi ini: performa ST-GCN sangat bergantung pada presisi lokasi keypoint karena noise pada koordinat sendi langsung mendistorsi struktur graph yang menjadi dasar spatial convolution, sementara LSTM dan TCN yang memperlakukan pose sebagai vektor flat relatif lebih toleran terhadap noise keypoint kecil.

Dibandingkan dengan Raza dkk. [12] yang melaporkan akurasi ST-GCN 98,44% tanpa analisis efisiensi komputasi, penelitian ini menunjukkan bahwa akurasi tinggi tersebut kemungkinan besar dicapai dengan biaya komputasi yang signifikan, trade-off yang tidak terlihat tanpa evaluasi efisiensi eksplisit sebagaimana dilakukan pada penelitian ini. Temuan ini menjawab pertanyaan penelitian kedua (RQ2): kompleksitas spatio-temporal ST-GCN tidak memberikan keunggulan signifikan dibandingkan pendekatan temporal murni (TCN) pada konteks deteksi jatuh dengan skeleton tunggal, sekaligus menegaskan pentingnya melaporkan efisiensi komputasi berdampingan dengan akurasi klasifikasi (RQ1) untuk memandu pemilihan arsitektur pada deployment dunia nyata.

4. KESIMPULAN

Penelitian ini membandingkan LSTM, TCN, dan ST-GCN untuk deteksi jatuh berbasis pose estimation pada dataset Le2i, dengan mengevaluasi performa klasifikasi dan efisiensi komputasi secara simultan. Empat kesimpulan utama dapat ditarik.

Pertama, menjawab pertanyaan penelitian pertama, TCN merupakan arsitektur yang paling seimbang antara akurasi dan efisiensi, dengan F1-Score 1,000 pada Lecture_room dan 0,875 (Recall 0,824) pada Office disertai efisiensi pelatihan yang kompetitif (2,97 detik/epoch); perlu dicatat bahwa keunggulan akurasi TCN atas LSTM dan ST-GCN tidak signifikan secara statistik (uji McNemar, Bagian 3.2), sehingga rekomendasi TCN terutama didasarkan pada keseimbangan trade-off akurasi-efisiensi, bukan superioritas akurasi tunggal. LSTM menjadi alternatif yang layak untuk perangkat dengan keterbatasan sumber daya yang ekstrem (inference 6,7× lebih cepat pada 0,276 ms/seq), namun dengan konsekuensi 29,4% kejadian jatuh pada Office terlewat (Recall 0,706) pada konfigurasi final.

Kedua, ekstraksi pose mendominasi latensi pipeline (~90%), dengan YOLOv11-pose memakan sekitar 27 ms per frame dibandingkan dengan 0,41–3,16 ms untuk klasifikasi. Ketiga model mencapai 16,2–17,9 FPS pada streaming webcam dengan konsumsi VRAM stabil 115 MB, mengonfirmasi kelayakan deployment real-time. Perbedaan inference time antar classifier memberikan dampak minimal terhadap FPS keseluruhan, sehingga optimasi latensi sistem ke depan sebaiknya difokuskan pada tahap ekstraksi pose. Temuan ini merupakan realisasi kontribusi kedua penelitian ini, yaitu analisis latensi end-to-end pada pipeline real-time yang memberikan panduan konkret prioritas optimasi untuk deployment nyata.

Ketiga, menjawab pertanyaan penelitian kedua, ST-GCN tidak memberikan keunggulan yang signifikan dibandingkan TCN pada konteks ini: ST-GCN tidak pernah mengungguli TCN pada skenario manapun (tidak ditemukan bukti keunggulan statistik ST-GCN atas TCN melalui uji McNemar, p=0,625), dan efisiensi parameternya (2,5× lebih sedikit parameter) tidak diterjemahkan menjadi kecepatan komputasi akibat operasi graph convolution yang kurang dapat diparalelkan.

Keempat, kualitas backbone memberikan dampak lebih besar daripada kompleksitas arsitektur: upgrade dari YOLOv11n ke YOLOv11m menghasilkan peningkatan performa terbesar, terutama untuk ST-GCN (F1-Score Office: 0,706 → 0,875), menunjukkan bahwa kualitas pose estimation merupakan faktor dominan untuk performa sistem. Temuan ini menegaskan nilai kontribusi metodologis penelitian ini yang secara eksplisit memisahkan pengaruh kualitas backbone pose estimation dari algoritma post-processing melalui empat tahap eksperimen terkontrol (Tabel 4).

Penelitian ini memiliki sejumlah keterbatasan yang perlu diperhatikan dalam menginterpretasikan hasil. Pertama, dataset Le2i berskala kecil (190 video) dengan satu subjek per video dan aktor bukan lansia sesungguhnya, sehingga validitas eksternal terhadap populasi lansia dengan dinamika gerakan berbeda masih perlu diverifikasi. Kedua, evaluasi dilakukan pada satu kali pembagian train-test dengan fixed random seed tanpa pengulangan (repeated run), sehingga variansi performa antar-run belum terkuantifikasi; uji signifikansi

statistik antar model pada split tunggal ini telah dilakukan (uji McNemar, Bagian 3.2), namun belum dapat mengonfirmasi apakah hasil ini konsisten pada split atau seed yang berbeda. Ketiga, parameter post-processing fall-streak accumulator menggunakan konfigurasi seragam untuk ketiga model meskipun terbukti berdampak berbeda per arsitektur (Bagian 3.2), sehingga performa ST-GCN yang dilaporkan berpotensi meningkat dengan tuning post-processing spesifik-model. Rekomendasi pengembangan berikut secara langsung merespons keterbatasan di atas serta temuan utama penelitian ini.

Penelitian selanjutnya dapat diarahkan pada lima aspek: (1) perluasan dataset pengujian dengan variasi ruangan yang lebih representatif terhadap kondisi pemantauan lansia seperti kamar tidur, kamar mandi, dan tangga, serta penggunaan subjek lansia sesungguhnya dengan postur dan dinamika gerakan yang berbeda; (2) optimasi tahap ekstraksi pose sebagai bottleneck dominan (~90% latensi) menggunakan model yang lebih ringan seperti MoveNet atau BlazePose, atau teknik akselerasi seperti quantization dan TensorRT; (3) pengembangan parameter post-processing yang disesuaikan secara spesifik per arsitektur, mengingat konfigurasi seragam terbukti merugikan ST-GCN meskipun membantu LSTM dan TCN; (4) validasi pemantauan kontinu jangka panjang (>24 jam) untuk memastikan stabilitas FPS, ketiadaan memory leak, dan keandalan sistem pada periode malam yang menjadi waktu rawan kejadian jatuh pada lansia; serta (5) evaluasi dengan multiple random seed atau repeated k-fold cross-validation untuk mengkuantifikasi variansi performa antar-run dan memverifikasi konsistensi hasil uji signifikansi statistik (uji McNemar, Bagian 3.2) pada split atau seed yang berbeda, menjawab keterbatasan kedua penelitian ini.

REFERENSI

[1] United Nations, “World Population Ageing 2019: Highlights,” UN DESA, 2019.

[2] World Health Organization, “Falls,” WHO, Apr. 2021. https://www.who.int/news-room/fact-sheets/detail/falls

[3] R. Kakara, G. Bergen, E. Burns, and M. Stevens, “Nonfatal and Fatal Falls Among Adults Aged ≥65 Years, United States, 2020–2021,” MMWR, vol. 72, no. 35, pp. 938–943, 2023, doi: 10.15585/mmwr.mm7235a1.

[4] Y. Wang and T. Deng, “Enhancing elderly care: Efficient and reliable real-time fall detection algorithm,” Digital Health, vol. 10, pp. 1–11, 2024, doi: 10.1177/20552076241233690.

[5] E. Alam, A. Sufian, P. Dutta, and M. Leo, “Vision-based human fall detection systems using deep learning: A review,” Comput. Biol. Med., vol. 146, p. 105626, 2022, doi: 10.1016/j.compbiomed.2022.105626.

[6] E. Tîrziu, A.-M. Vasilevschi, A. Alexandru, and E. Tudora, “Real-Time Fall Monitoring for Seniors via YOLO and Voice Interaction,” Future Internet, vol. 17, no. 8, p. 324, 2025, doi: 10.3390/fi17080324.

[7] F. Shu and J. Shu, “An eight-camera fall detection system using human fall pattern recognition via machine learning by a low-cost android box,” Sci. Rep., vol. 11, no. 1, p. 2471, 2021, doi: 10.1038/s41598-021-81115-9.

[8] A. Sherstinsky, “Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network,” Physica D, vol. 404, p. 132306, 2020, doi: 10.1016/j.physd.2019.132306.

[9] S. Bai, J. Z. Kolter, and V. Koltun, “An empirical evaluation of generic convolutional and recurrent networks for sequence modeling,” arXiv:1803.01271, 2018. [Online]. Available: https://arxiv.org/abs/1803.01271

[10] S. Yan, Y. Xiong, and D. Lin, “Spatial temporal graph convolutional networks for skeleton-based action recognition,” in Proc. AAAI, vol. 32, no. 1, pp. 7444–7451, 2018, doi: 10.1609/aaai.v32i1.12328.

[11] M. Salimi, J. J. M. Machado, and J. M. R. S. Tavares, “Using deep neural networks for human fall detection based on pose estimation,” Sensors, vol. 22, no. 12, p. 4544, 2022, doi: 10.3390/s22124544.

[12] A. Raza, M. H. Yousaf, W. Ahmad, S. A. Velastin, and S. Viriri, “Human fall detection using pose estimation: From traditional machine learning to vision transformers,” Eng. Appl. Artif. Intell., vol. 143, p. 109809, 2025, doi: 10.1016/j.engappai.2024.109809.

[13] I. Charfi, J. Miteran, J. Dubois, M. Atri, and R. Tourki, “Optimized spatio-temporal descriptors for real-time fall detection: comparison of support vector machine and Adaboost-based classification,” J. Electron. Imaging, vol. 22, no. 4, p. 041106, 2013, doi: 10.1117/1.JEI.22.4.041106.

[14] R. Khanam and M. Hussain, “YOLOv11: An overview of the key architectural enhancements,” arXiv:2410.17725, 2024. [Online]. Available: https://arxiv.org/abs/2410.17725

[15] T.-Y. Lin et al., “Microsoft COCO: Common objects in context,” in ECCV 2014, pp. 740–755, Springer, 2014, doi: 10.1007/978-3-319-10602-1_48.

[16] O. Rainio, J. Teuho, and R. Klén, “Evaluation metrics and statistical tests for machine learning,” Sci. Rep., vol. 14, p. 6086, 2024, doi: 10.1038/s41598-024-56706-x.

[17] L. Qi and H. Sun, “Multistage fall detection framework via 3D pose sequences and TCN integration,” Sci. Rep., vol. 15, p. 27832, 2025, doi: 10.1038/s41598-025-11325-y.

[18] F. Castro, V. Dentamaro, V. Gattulli, and D. Impedovo, “Fall detection with LSTM and attention mechanism,” in Proc. WAMWB@MobileHCI 2023, CEUR Workshop Proceedings, vol. 3517, pp. 37–50, 2023.
[19] J. Shin, A. S. M. Miah, R. Egawa, K. Hirooka, M. A. M. Hasan, Y. Tomioka, and Y. S. Hwang, “Fall recognition using a three stream spatio temporal GCN model with adaptive feature aggregation,” Sci. Rep., vol. 15, p. 10635, 2025, doi: 10.1038/s41598-025-95508-7.
[20] Y. Cho, G. Solak, O. Nocentini, M. Lorenzini, A. Fortuna, and A. Ajoudani, “Anticipatory Fall Detection in Humans with Hybrid Directed Graph Neural Networks and Long Short-Term Memory,” in Proc. IEEE RO-MAN, 2025, arXiv:2509.05337.
[21] M. Zobi, L. Bolzani, Y. Tabii, and R. Oulad Haj Thami, “Robust 3D Skeletal Joint Fall Detection in Occluded and Rotated Views Using Data Augmentation and Inference-Time Aggregation,” Sensors, vol. 25, no. 21, p. 6783, 2025, doi: 10.3390/s25216783.
[22] J. Yan, X. Wang, J. Shi, and S. Hu, “Skeleton-Based Fall Detection with Multiple Inertial Sensors Using Spatial-Temporal Graph Convolutional Networks,” Sensors, vol. 23, no. 4, p. 2153, 2023, doi: 10.3390/s23042153.