CPSM | Conference Proceedings in Science and Management

Proceedings of the 2nd International Engineering Data Analytics and Management Conference (EAMCON 2026), 8 to 10 July 2026, Shinawatra University, Thailand

Published: October 2026
DOI: https://doi.org/10.68337/cpsm.v1.i2.2026-025

Vol. 1, Issue 2 | Proceeding Paper

A Context-Aware Neural Retrieval Framework for Fall Detection in IoT-Based Elderly Care

Vanaj Kalyan S. R.[1,*], Dharshan S. S.[1], R. S. Pavithra[2], Puja Sai Akshath[2], and Nihal Gopisetti[2]

[1]*Sri Eshwar College of Engineering, Coimbatore, Tamil Nadu, India*

[2]*Anurag University, Hyderabad, Telangana, India*

[*]Corresponding author: iamvanaj@gmail.com

Abstract. Wearable Internet of Things (IoT) devices produce long, continuous streams of inertial data, and the difficult part is picking out the few moments that matter. Most fall detection systems treat each segment of data as a separate yes-or-no decision. That works in a laboratory, but it gives a caregiver no help in deciding which of 50 alerts to check first. We approach fall monitoring as an information retrieval problem instead. Accelerometer and gyroscope signals are split into overlapping windows, each window is encoded with a transformer, and the system learns to rank windows by how likely they are to contain a fall. On top of the ranking we add two context signals, one measuring motion instability and one estimating impact severity, so that the most dangerous events come out on top. On the SisFall dataset the system reaches an average precision (AP) of 0.983 with a normalized discounted cumulative gain at rank 10 (nDCG@10) of 1.0, so real fall events almost always appear at the top of the ranked list, ahead of convolutional neural network (CNN), long short-term memory (LSTM), and transformer classification baselines. The result is a monitoring system that gives ranked, interpretable output instead of a stream of binary alarms and that matches how caregivers handle events in practice.

Keywords. Fall detection; neural information retrieval; learning to rank; IoT sensor analytics; transformer models; intelligent systems

I. Introduction

Falls are one of the main reasons elderly people are admitted to hospital, and the usual response is some form of continuous monitoring [1]. Monitoring based on artificial intelligence (AI) has proven useful in domains such as flood forecasting [2], air quality prediction [3], and mental health screening [4], and elderly care is among the highest-stakes settings. Low-cost wearable sensors make it easy to record accelerometer and gyroscope data around the clock, and much work has gone into deciding whether a given segment of these data contains a fall [5], [6]. The earliest systems used simple acceleration thresholds [7], later ones moved to classical machine learning, and the current generation uses deep networks [8].

The problem is that almost all of this work stops at the binary decision. A deployed system has to handle hours of streaming data and hundreds of candidate events per day, while a caregiver can check only a few of them. A classifier with 95% accuracy still floods staff with alerts when run continuously, with no principled way to decide which alert matters most. The result is alarm fatigue: when everything beeps, people stop listening.

Viewed this way, the problem is one of information retrieval (IR): a large collection of items (sensor windows), few of them relevant (actual falls), and a user who wants the relevant ones ranked at the top [9]. The IR community has mature training objectives [10], evaluation metrics [11], and strong neural ranking models [12], [13] for this kind of problem, yet almost none of it has reached sensor-based health monitoring.

So instead of another classifier, we built a retrieval system. Each window of sensor data is treated like a document, a transformer encoder [14] turns it into an embedding, and a scoring function ranks all windows by fall relevance. Two extra signals, motion instability and estimated injury severity, then re-rank the list so that the most urgent events surface first. We make four contributions. We formulate fall monitoring as neural information retrieval over continuous sensor streams, which to our knowledge is new in this setting. We build a transformer-based learning-to-rank model for it. We add a context-aware prioritization step on top of the ranking. And we evaluate the system on SisFall [15], where it clearly outperforms classification baselines on the ranking metrics.

II. Related Work

A. Fall Detection With Wearable Sensors

Early systems used fixed thresholds on acceleration magnitude. The algorithm of Bourke, O'Brien, and Lyons [7] is a representative example, and such methods are still common in commercial pendant devices. Their weakness is that thresholds do not generalize well, since fast sitting or jumping can cross them too. The next wave applied classical machine learning to handcrafted features, which surveys from

Copyright © 2026 The Authors. Published by SHORAM Editorial Services, India, under the Creative Commons Attribution 4.0 License (CC BY 4.0).

that period cover well [1], [5]. Public datasets enabled systematic comparison: MobiFall [16], UMAFall [17], and SisFall [15], which we use because it includes elderly participants and a broad range of fall types. Deep learning then replaced feature engineering. Ordóñez and Roggen [8] combined convolutional and long short-term memory (LSTM) layers to learn strong representations directly from raw wearable signals, and Wang, Ellul, and Azzopardi [6] surveyed the fall detection literature that followed. Across all of this work, though, each window receives an independent yes-or-no decision, and nothing ranks or prioritizes events across a recording. That gap is the one this work targets.

B. Transformers for Time-Series Data

Transformers [14] were built for language but have spread to almost every sequence domain, including time series. For inertial data the appeal is that self-attention can connect the short, sharp pattern of an impact with the longer context around it, without handcrafted features. We use a transformer only as an encoder inside a retrieval pipeline.

C. Neural IR and Learning to Rank

Liu's survey [9] and RankNet [10] cover the learning-to-rank foundations we borrow, and normalized discounted cumulative gain (nDCG) [11] provides evaluation for ranked output. On the neural side, transformer encoders changed text ranking through neural re-ranking [12] and dense retrieval [13], though strong lexical baselines still outperform neural rankers on some retrieval tasks [18]. Our system transplants this neural stack to sensor data, which to our knowledge has not been done for fall monitoring before. It also differs from ranking-based anomaly detection. Anomaly detectors score deviation from learned normal behavior without supervision, whereas our ranker is trained on labeled fall relevance and then re-ranked with task-specific context signals, which is a different objective with a different output.

III. System Overview

Fig. 1 shows the pipeline. Inertial streams are cut into overlapping windows, each a candidate document. A transformer encoder produces an embedding and a relevance score for each window, a cloud retrieval engine indexes these and returns the top-ranked candidates, re-ranking combines relevance with the instability and severity signals, and the prioritized list goes to the care provider, who receives ranked events instead of isolated yes-or-no decisions.

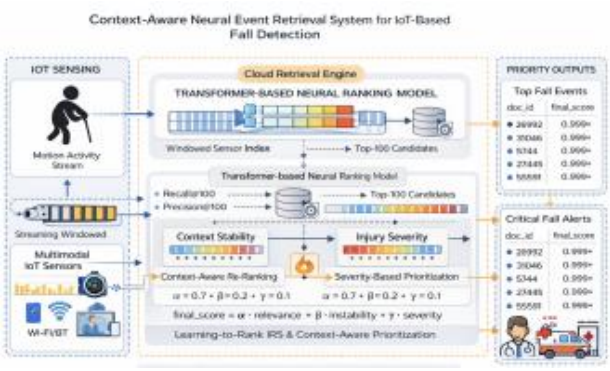

Fig. 1. Context-aware neural event retrieval system for IoT-based fall detection.

IV. Methodology

A. Retrieval Formulation

Let $X = \{x_t\}$, $t = 1, \ldots, T$, be a continuous multivariate sensor stream, where each $x_t \in \mathbb{R}^d$ holds synchronized accelerometer and gyroscope readings at time t. We segment the stream into overlapping fixed-length windows $W = \{w_i\}$, $i = 1, \ldots, N$, each with six channels (three acceleration and three angular velocity); the overlap ensures that a fall near a window boundary is not lost. Rather than classifying each w_i, we treat it as a retrievable document and learn a ranking function $f(w_i)$ that scores fall windows above windows of normal activities of daily living (ADLs). The output is a ranking over the whole stream rather than a set of independent labels.

B. Transformer Encoder

Each window passes through stacked multi-head self-attention layers with feed-forward projections. Attention suits falls because the evidence is spread out in time: a brief free-fall phase, a sharp impact spike, and then an unusually still period. A convolution with a small receptive field sees these pieces separately, whereas attention connects them, and multiple heads can associate an acceleration spike with simultaneous gyroscope rotation. We deliberately kept the encoder standard [14]. The contribution here is the retrieval framing, not a new architecture, so the gains should come from the formulation rather than from architectural changes.

C. Training Objective

Binary cross-entropy works but optimizes each window independently and says nothing about ordering. Since the goal is a ranking, we also use a pairwise margin loss in the spirit of RankNet [10]. For a fall window w^+ and a non-fall window w^- sampled from the same stream,

$$\mathcal{L}_{\text{rank}} = \max\left(0,\ m - S_r(w^+) + S_r(w^-)\right), \qquad (1)$$

where m is a fixed margin. Same-stream sampling forces the model to separate falls from that person's own normal movements, which is the comparison a deployed system makes. Table 1 shows that the pairwise loss improves every metric.

Table 1. Effect of Training Objective on Retrieval Performance

Training loss	AP	nDCG@10	MRR
Binary cross-entropy	0.963	0.971	0.968
Pairwise ranking loss	**0.983**	**1.000**	**1.000**

D. Scoring and Context-Aware Re-Ranking

A neural scoring head maps each embedding to a continuous relevance score S_r. Ranking by S_r already gives confidence-aware ordering, top-k alerting, and direct compatibility with IR metrics. But relevance alone does not capture urgency: two falls can both be real while one is far more dangerous. We therefore compute an instability score S_i, which measures motion irregularity through sudden variance changes across channels, and an injury severity score S_s, which estimates impact severity from signal magnitude and temporal consistency, with a large spike followed by sustained stillness as the worst case. The final score is

$$S = \alpha S_r + \beta S_i + \gamma S_s, \tag{2}$$

where α, β, and γ balance the three signals and act as interpretable controls: a facility that must never miss a severe event can raise γ. In our experiments we set $\alpha = 0.7$, $\beta = 0.2$, and $\gamma = 0.1$ manually; a systematic sensitivity analysis of these weights is left for future work. S_i and S_s are signal-level heuristics for triage ordering, not medical assessments.

E. Deployment and Usage

Segmentation is linear in stream length, and the fixed window length keeps the inference cost per window constant despite the quadratic cost of transformer attention. Light preprocessing runs on the edge device, while encoding and ranking run in the cloud, all on central processing units (CPUs), so no accelerators are needed. Staff see a list sorted by S with severe falls on top, which keeps a human in the loop for every decision.

V. Experimental Setup and Results

A. Setup

We used SisFall [15], which contains recordings of ADLs and several types of simulated falls from young and elderly subjects, and chose it over MobiFall [16] and UMAFall [17] for its elderly participants and range of fall types. Recordings were segmented into overlapping six-channel windows, labeled by dominant activity, and treated as independent retrieval units. Because the system outputs a ranking, we evaluated it with ranking metrics rather than accuracy: average precision (AP) over the full ranked list, mean reciprocal rank (MRR) for how early the first true fall appears, and nDCG [11] at cutoffs 10 and 50, which mimic realistic alerting budgets. The model was implemented in PyTorch and trained for 25 epochs with the Adam optimizer at standard settings. We ran all experiments on CPUs, since that is the hardware most IoT healthcare deployments have.

B. Main Results

Table 2 gives the overall results: an AP of 0.983, with MRR and nDCG at 1.000, so fall events land at the top of the ranking almost every time. For emergency response, the MRR of 1.0 matters most: in our test runs, the first event the caregiver saw was always a real fall. We read these near-perfect values with some caution. Simulated falls have very distinctive impact signatures, so once the encoder has learned them, the ranking task on SisFall is nearly saturated, and the scores should be taken as an upper bound rather than a prediction of field performance.

Table 2. Neural Retrieval Performance on the SisFall Dataset

Metric	Value
Average precision (AP)	0.983
Mean reciprocal rank (MRR)	1.000
nDCG@10	1.000
nDCG@50	1.000

Table 3 compares classification baselines whose scores are used directly as rankings for a fair evaluation. Their ordering follows the expected pattern, with the transformer classifier ahead of the LSTM and CNN, and our model outperforms all three. The gap to the transformer classifier (0.983 versus 0.945 AP) is the most informative, since the encoder is the same; the difference comes only from training and using it as a ranker.

Table 3. Comparison With Baseline Fall Detection Models

Method	AP	nDCG@10
CNN-based classifier	0.912	0.891
LSTM-based classifier	0.928	0.904
Transformer classifier	0.945	0.926
Proposed retrieval model	**0.983**	**1.000**

C. Ablation and Score Distributions

Table 4 isolates the re-ranking signals. Relevance alone gives an AP of 0.941, instability re-ranking raises it to 0.971, and adding severity gives the full 0.983 with an nDCG@10 and MRR of 1.000. The severity signal in particular pushes the most critical events to the top consistently; without it, a dramatic but harmless movement occasionally outranked a dangerous fall.

Fig. 2 plots the distribution of final priority scores for retrieved fall events. The scores cluster tightly near 1.0 while the ordering among them is preserved, which is what monitoring needs. A caregiver does not need calibrated probabilities, only event A ranked above event B whenever A is more urgent.

Table 4. Ablation Study of Context-Aware Re-Ranking Components

Model variant	AP	nDCG@10	MRR
Transformer (relevance only)	0.941	0.923	0.912
+ Instability re-ranking	0.971	0.981	0.979
+ Severity (full model)	**0.983**	**1.000**	**1.000**

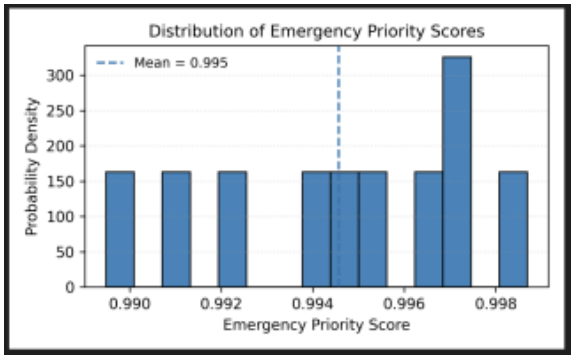

Fig. 2. Distribution of emergency priority scores for retrieved fall events (mean 0.995).

VI. Discussion

The main advantage of the retrieval framing is that it gives more than classification at almost no extra cost. The same encoder that would have driven a classifier drives a ranker instead, and the output becomes ordered, confidence-aware, and re-rankable by urgency, without extra sensors. The failure cases are mostly transitional movements that look like falls in the raw signal, such as sitting down abruptly or stumbling without impact. This is the usual trade-off between sensitivity and specificity, but the ranked output makes it manageable: each facility can choose how many top-ranked events to review per shift and set its cutoff accordingly.

The work has clear limits. All experiments use a single benchmark, and the falls in SisFall are simulated onto mattresses, so the numbers are an upper bound on real-world performance; validation on MobiFall and UMAFall is the immediate next step. The results come from single runs without confidence intervals or significance tests, and the baselines are classification models, so comparisons with ranking-based anomaly detection and time-series retrieval methods are still needed to position the work fully. The S_i and S_s scores are engineering quantities rather than clinical assessments, and the model is not personalized even though movement patterns differ considerably between individuals. The caregiver triage benefits argued for here have not yet been tested in a user study, and inference latency and memory have not yet been measured on deployment hardware. These points define our future work: evaluation on several datasets with repeated runs, broader baselines, per-user risk modeling, learned re-ranking weights, deployment measurements, and a caregiver-in-the-loop study.

The system is meant for decision support only. Intervention decisions stay with the caregiver, and the ranked output helps by being transparent: several candidate events, and the contribution of each signal (S_r, S_i, and S_s) to their scores, can be inspected instead of one opaque binary alarm being trusted. Richer explanations, such as attention visualizations over the sensor window, would strengthen this further and are part of our planned work.

VII. Conclusion

We have presented a context-aware neural retrieval system for IoT-based fall detection. The central step is to rank windows instead of classifying them. A transformer encoder with a learning-to-rank objective retrieves fall events from continuous streams, and instability and severity signals re-rank them by urgency. On SisFall the system reached an AP of 0.983 with an nDCG@10 and MRR of 1.000, ahead of CNN, LSTM, and transformer baselines. The ranked output matches how monitoring works in practice, runs on CPU-class hardware, and should transfer to other continuous monitoring tasks such as seizure or arrhythmia detection.

Author Contributions

All authors contributed to the conception and preparation of this work and approved the final version of the manuscript.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Declaration of Generative AI Use

The authors declare that no generative artificial intelligence tools were used in the preparation of this manuscript. The authors take full responsibility for the content of the publication.

Data Availability

This study used the publicly available SisFall dataset [15]. Further data supporting the findings are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] Raul Igual, Carlos Medrano, and Inmaculada Plaza, "Challenges, issues and trends in fall detection systems," *BioMed. Eng. OnLine*, vol. 12, no. 1, Art. no. 66, 2013, doi: 10.1186/1475-925X-12-66.

[2] Amir Mosavi, Pinar Ozturk, and Kwok-wing Chau, "Flood prediction using machine learning models: Literature review," *Water*, vol. 10, no. 11, Art. no. 1536, 2018, doi: 10.3390/w10111536.

[3] Manuel Méndez, Mercedes G. Merayo, and Manuel Núñez, "Machine learning algorithms to forecast air quality: A survey," *Artif. Intell. Rev.*, vol. 56, no. 9, pp. 10031-10066, 2023, doi: 10.1007/s10462-023-10424-4.

[4] Adrian B. R. Shatte, Delyse M. Hutchinson, and Samantha J. Teague, "Machine learning in mental health: A scoping review of methods and applications," *Psychol. Med.*, vol. 49, no. 9, pp. 1426-1448, 2019, doi: 10.1017/S0033291719000151.

[5] Muhammad Mubashir, Ling Shao, and Luke Seed, "A survey on fall detection: Principles and approaches," *Neurocomputing*, vol. 100, pp. 144-152, 2013, doi: 10.1016/j.neucom.2011.09.037.

[6] Xueyi Wang, Joshua Ellul, and George Azzopardi, "Elderly fall detection systems: A literature survey," *Front. Robot. AI*, vol. 7, Art. no. 71, 2020, doi: 10.3389/frobt.2020.00071.

[7] A. K. Bourke, J. V. O'Brien, and G. M. Lyons, "Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm," *Gait Posture*, vol. 26, no. 2, pp. 194-199, 2007, doi: 10.1016/j.gaitpost.2006.09.012.

[8] Francisco Ordóñez and Daniel Roggen, "Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, Art. no. 115, 2016, doi: 10.3390/s16010115.

[9] Tie-Yan Liu, "Learning to rank for information retrieval," *Found. Trends Inf. Retr.*, vol. 3, no. 3, pp. 225-331, 2009, doi: 10.1561/1500000016.

[10] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender, "Learning to rank using gradient descent," in *Proc. 22nd Int. Conf. Mach. Learn. (ICML)*, 2005, pp. 89-96, doi: 10.1145/1102351.1102363.

[11] Kalervo Järvelin and Jaana Kekäläinen, "Cumulated gain-based evaluation of IR techniques," *ACM Trans. Inf. Syst.*, vol. 20, no. 4, pp. 422-446, 2002, doi: 10.1145/582415.582418.

[12] Rodrigo Nogueira and Kyunghyun Cho, "Passage re-ranking with BERT," arXiv:1901.04085, 2019, doi: 10.48550/arXiv.1901.04085.

[13] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih, "Dense passage retrieval for open-domain question answering," in *Proc. 2020 Conf. Empirical Methods Nat. Lang. Process. (EMNLP)*, 2020, pp. 6769-6781, doi: 10.18653/v1/2020.emnlp-main.550.

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 5998-6008.

[15] Angela Sucerquia, José David López, and Jesús Francisco Vargas-Bonilla, "SisFall: A fall and movement dataset," *Sensors*, vol. 17, no. 1, Art. no. 198, 2017, doi: 10.3390/s17010198.

[16] George Vavoulas, Matthew Pediaditis, Emmanouil G. Spanakis, and Manolis Tsiknakis, "The MobiFall dataset: An initial evaluation of fall detection algorithms using smartphones," in *Proc. 13th IEEE Int. Conf. Bioinform. Bioeng. (BIBE)*, 2013, pp. 1-4, doi: 10.1109/BIBE.2013.6701629.

[17] Eduardo Casilari, José A. Santoyo-Ramón, and José M. Cano-García, "UMAFall: A multisensor dataset for the research on automatic fall detection," *Procedia Comput. Sci.*, vol. 110, pp. 32-39, 2017, doi: 10.1016/j.procs.2017.06.110.

[18] Praveen Kumar, "When lexical beats neural: A content-aware analysis of duplicate bug report retrieval across software projects," *Preprints*, 2026, doi: 10.20944/preprints202606.1069.v1.