Fall Detection for LLMs

Himani Shah, Chelsea Jaculina

Department of Applied Data Science
San Jose State University

***Abstract*—The study presents a fall detection system that combines both video and sensor data using Large Language Models (LLMs). Traditional systems often rely only on wearables or cameras, which can be uncomfortable or raise privacy concerns. Our approach uses a mix of prompting techniques—like Tree-of-Thought (ToT), Chain-of-Thought (CoT), and One/Few/Zero-Shot examples—along with Retrieval-Augmented Generation (RAG) and lightweight fine-tuning using LoRA and Reinforcement Learning from Human Feedback (RLHF). We tested the system using the UR Fall Detection Dataset and found that ToT prompting and LoRA + RLHF fine-tuning gave the best results, with high accuracy and strong F1 scores. This system not only detects falls effectively in real-world settings but also shows promise for future use on lightweight, portable devices for real-time monitoring.**

***Index Terms*—Fall Detection, Multimodal Learning, Large Language Models (LLMs), Prompt Engineering, Tree-of-Thought (ToT), Chain-of-Thought (CoT), Few-Shot Learning, Zero-Shot Learning, Retrieval-Augmented Generation (RAG), Parameter-Efficient Fine-Tuning (PEFT), LoRA, Reinforcement Learning from Human Feedback (RLHF), UR Fall Detection Dataset**

I. INTRODUCTION

Falls are among the most serious health risks. As a result, there is a growing need for fall detection systems that are not only dependable but also sensitive to the privacy and comfort individuals. Traditional methods, rely on using surveillance cameras or using or wearable devices, which can be uncomfortable or intrusive, complicating their use in private or home environments (Mubashir, 2013).

The development of artificial intelligence has given us new methods to create more intelligent, human-like fall detection systems, particularly through the use of Large Language Models (LLMs) and the combination of various data kinds. However, many current systems still face problems when used in real-world situations, where things like body movement, lighting, and background conditions can vary a lot (Naser, 2022).

In this study, we introduce a complete fall detection system that combines video and sensor data using advanced prompting and training methods. We convert sensor and video signals into a text-like format so the system can better understand and reason about what's happening—similar to how a person might think. We apply prompting strategies like Tree-of-Thought (ToT) (Yao, 2023), Chain-of-Thought (CoT) (Wei, 2022), and Few/One/Zero-Shot learning . We also improve the system's accuracy and performance using techniques like Retrieval-Augmented Generation (RAG) (Lewis, 2020), LoRA for lightweight fine-tuning, and Reinforcement Learning from Human Feedback (RLHF) (Pillai, 2023).

We tested our system using the UR Fall Detection Dataset (Kwokek and Kepski, 2023) and found that it performs very well, especially with ToT and LoRA+RLHF, giving strong results with high F1-scores. Overall, our system offers a practical and explainable solution for detecting falls in different environments, and it's efficient enough to be used on portable devices for real-time monitoring—making it both helpful and privacy-friendly.

II. RELATED WORK

Fall detection research has traditionally relied on threshold-based heuristics and conventional machine learning classifiers, often constrained by hand-crafted features and a lack of contextual awareness. Early systems using wearable sensors or static vision models showed moderate success in constrained environments but consistently suffered from high false-positive rates in real-world scenarios (Kwolek & Kepski, 2014). These limitations have prompted the shift toward integrating Large Language Models (LLMs) and prompt engineering strategies to enable more human-like reasoning over multimodal data.

Foundational prompting techniques such as Zero-Shot, One-Shot, and Few-Shot learning were first established in large-scale autoregressive models (Brown et al., 2020), allowing models to generalize tasks with minimal training data. However, these methods often fall short in complex reasoning tasks such as activity recognition, where sequential logic and ambiguity handling are essential. To address these gaps, Chain-of-Thought (CoT) prompting introduced step-wise reasoning that enables LLMs to decompose and solve multi-step problems (Wei et al., 2022), while Tree-of-Thought (ToT) prompting extended this by exploring multiple reasoning paths in parallel and backtracking when necessary (Yao et al., 2023). These techniques have shown measurable improvements in domains requiring judgment under uncertainty, including algorithmic problem solving and decision-making simulations.

In the context of activity recognition, such structured prompting enables nuanced classification by modeling not just the outcome but also the thought process leading to it.For example, Xie et al. (2023) showed that ToT-enabled models outperformed baseline CoT methods in video reasoning benchmarks, showing stronger robustness to temporal disruptions and occlusions. Furthermore, the integration of Retrieval-Augmented Generation (RAG) (Lewis et al., 2020) allows the model to reference relevant examples during inference, enhancing contextual grounding, while Parameter-Efficient Fine-Tuning (PEFT) methods like LoRA (Hu et al., 2021) allow the adaptation of large models with minimal compute overhead.

Recent research has begun to investigate the effectiveness of such approaches in safety-critical applications. Zhang et al. (2024) used CoT prompting for human activity understanding in healthcare monitoring scenarios and reported a 14% reduction in misclassification errors compared to flat classification pipelines. Wang et al. (2023) used LoRA-tuned LLMs in smart home fall detection systems to achieve near-real-time inference speeds on edge devices without sacrificing accuracy. These developments highlight the growing promise of combining multi-shot prompting, semantic retrieval, and reward-based tuning to create fall detection systems that are not only accurate, but also interpretable and adaptable to real-world variability.

Unlike earlier fall detection systems dependent on handcrafted rules or black-box deep networks, our framework combines multimodal prompting, semantic encoding, and reinforcement-aligned outputs to bridge these performance gaps. By transforming sensor and video inputs into natural language–like prompts, and applying structured reasoning via CoT and ToT, our system simulates expert-level decision-making with explainability and reliability at its core.

Recent advances in Retrieval-Augmented Generation (RAG) have shown potential for enhancing fall detection systems by integrating contextual knowledge during inference. Ji et al. (2024) explore how RAG-based architectures can support activity recognition tasks, including fall detection, by leveraging external knowledge bases to improve decision-making accuracy. Despite increased computational costs, their findings suggest that the retrieval mechanism helps disambiguate challenging postural transitions and improves interpretability in fall scenarios.

In the medical domain, Pillai et al. (2023) demonstrate how large language models with retrieval capabilities can extract fall-related incidents from clinical notes. Their zero-shot model effectively identifies documented postoperative falls, achieving high precision and recall. This illustrates how RAG-style augmentation can boost performance in medical fall-detection applications involving unstructured data.

Alkhalaf et al. (2023) present a healthcare summarization system using Llama-2 enhanced with RAG to reduce hallucinations and improve reliability. Though their work is not solely focused on fall detection, it shows that RAG can strengthen generative medical AI systems. Their architecture improves factual accuracy, which could be repurposed to support evidence-grounded fall alert generation in clinical workflows.

Wang et al. (2024) introduce a radar-based fall detection model named SelaFD, which uses Low-Rank Adaptation (LoRA) to efficiently fine-tune a Vision Transformer. Instead of full model retraining, they insert lightweight trainable rank-decomposed layers, preserving performance while reducing memory usage. Their LoRA-based model achieves superior results on standard radar datasets, supporting LoRA's effectiveness for activity-based fall detection.

Oraki et al. (2024) address the challenge of computationally expensive Transformer architectures in skeletal human activity recognition by introducing LORTSAR, a low-rank decomposed model. This architecture, though not exclusive to fall detection, demonstrates how LoRA-like parameter compression strategies allow adaptation of large models for fall-related action recognition in constrained environments.

Li et al. (2024) propose TrustLoRA, an extension of LoRA for detecting AI system failures using multiple low-rank adapters. While applied to anomaly detection beyond fall cases, the architecture's modular and controllable nature highlights LoRA's promise for personalized fall anomaly detection, especially when trained on user-specific motion deviations or environmental noise.

Naser et al. (2022) implement a human-in-the-loop fall detection system with thermal sensors, where AI-flagged alerts are validated by a caregiver. This real-time human feedback loop reduces false positives while maintaining high sensitivity. Though not termed RLHF, their method mirrors its principles, showing how interactive feedback can be harnessed to refine fall detection outcomes.

Villar et al. (2019) propose a personalized fall detection framework that adapts to each user's activity profile using real-time feedback. By integrating online learning with supervised corrections from caregivers or users, the system progressively reduces errors. This adaptive mechanism embodies reinforcement learning from human feedback in a fall detection setting.

Dias et al. (2023) present a smartwatch-based fall detector that actively requests feedback after suspected falls. Confirmed false positives are used to retrain the system, reducing alarm fatigue over time. This user-in-the-loop learning system effectively implements RLHF principles to enhance fall detection personalization.

III. DATASET

A. UR Fall Detection Dataset

This project utilizes the UR Fall Detection Dataset, a benchmark dataset created by Kepski and Kwolek at the University of Rzeszow. The dataset is designed for the development and evaluation of fall detection systems, particularly in contexts where privacy, multimodality, and robustness to environmental variability are critical. It provides synchronized RGB, depth, and accelerometer recordings of falls and daily activities performed in a controlled indoor setting.

Dataset Structure: The UR Fall Detection Dataset is composed of:

- **70 fall events** recorded from 6 individuals simulating various fall types, including forward, backward, lateral, and collapsing falls (e.g., fainting).
- **33 Activities of Daily Living (ADLs)**, which include sitting, walking, crouching, lying down, and other movements that might visually resemble falls.
- **Three synchronized data modalities**:

- **RGB videos** recorded at 25 FPS, 640×480 resolution.
- **Depth videos** captured at 30 FPS, using a Microsoft Kinect sensor.
- **Accelerometer data** collected using a 3-axis sensor worn at the waist.

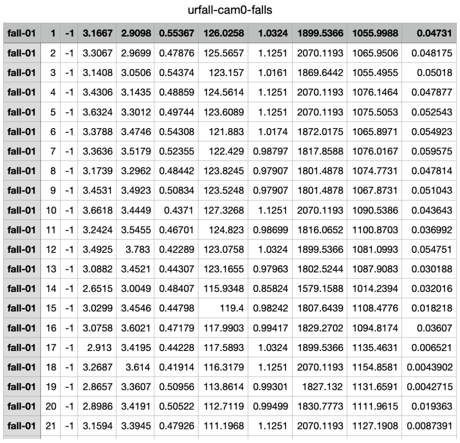
Each recording session is timestamped to enable multimodal alignment between video frames and acceleration signals. The RGB and depth video sequences are stored as AVI files, and the accelerometer data is stored in CSV format, with each row representing a timestamped reading of X, Y, and Z-axis acceleration values. The dataset is further organized into individual subdirectories by activity type and sequence index.

Data Diversity and Use Cases: A key strength of the UR Fall Dataset lies in its realistic representation of fall-like scenarios. The ADL actions are purposefully included to test the system's ability to distinguish between actual falls and non-fall activities with similar motion patterns. The fall actions are performed with different orientations, fall speeds, and recovery attempts, which contributes to variability in body posture and camera perspectives. This diversity allows for robust evaluation of detection models in challenging environments.

Extracted Features: To support fall detection tasks, the UR Fall Detection Dataset includes preprocessed CSV files—`urfall-cam0-falls.csv` and `urfall-cam0-adls.csv`—that contain feature vectors extracted from depth maps. Each row in these files corresponds to a single depth image frame and encapsulates numerical descriptors derived from the segmented human figure. The first two columns include the sequence name (e.g., `fall-01`, `adl-01`) and the frame number within the sequence. A posture `label` column follows, indicating whether the person is lying on the ground (`1`), not lying (`-1`), or in a transitional "falling" state (`0`); the latter is excluded during classification to avoid ambiguity.

The remaining columns represent a variety of shape and spatial distribution metrics:

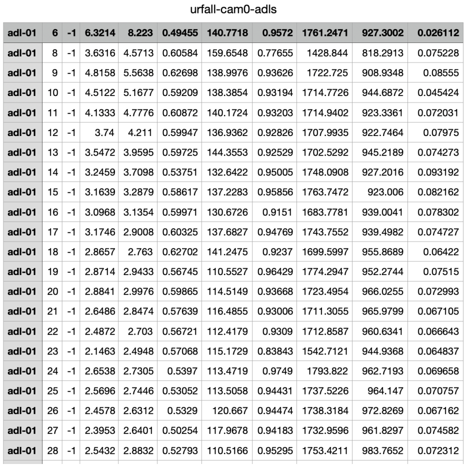
- **HeightWidthRatio**: Ratio of bounding box height to width, indicating posture orientation.
- **MajorMinorRatio**: Ratio of major to minor axes from the person's contour blob, capturing elongation characteristics.
- **BoundingBoxOccupancy**: Proportion of the bounding box occupied by the human silhouette.
- **MaxStdXZ**: Maximum standard deviation of pixels from the centroid along the horizontal (X) and depth (Z) axes, derived from the 3D point cloud.
- **HHmaxRatio**: Ratio of the person's current height to their known standing height, signaling collapse or lying posture.
- **H**: Estimated human height in millimeters.
- **D**: Distance from the person's center of mass to the floor, in millimeters.
- **P40**: Ratio of point cloud density within a 40cm floor-aligned cuboid to the density in a cuboid approximating the person's full height.



urfall-cam0-falls

fall-01	1	-1	3.1667	2.9098	0.55367	126.0258	1.0324	1899.5366	1055.9988	0.04731
fall-01	2	-1	3.3067	2.9699	0.47876	125.5657	1.1251	2070.1193	1065.9506	0.048175
fall-01	3	-1	3.1408	3.0506	0.54374	123.157	1.0161	1869.6442	1055.4955	0.05018
fall-01	4	-1	3.4306	3.1435	0.48859	124.5614	1.1251	2070.1193	1076.1464	0.047877
fall-01	5	-1	3.6324	3.3012	0.49744	123.6089	1.1251	2070.1193	1075.5053	0.052543
fall-01	6	-1	3.3788	3.4746	0.54308	121.883	1.0174	1872.0175	1065.8971	0.054923
fall-01	7	-1	3.3636	3.5179	0.52355	122.429	0.98797	1817.8588	1076.0167	0.059575
fall-01	8	-1	3.1739	3.2962	0.48442	123.8245	0.97907	1801.4878	1074.7731	0.047814
fall-01	9	-1	3.4531	3.4923	0.50834	123.5248	0.97907	1801.4878	1067.8731	0.051043
fall-01	10	-1	3.6618	3.4449	0.4371	127.3268	1.1251	2070.1193	1090.5386	0.043643
fall-01	11	-1	3.2424	3.5455	0.46701	124.823	0.98699	1816.0652	1100.8703	0.036992
fall-01	12	-1	3.4925	3.783	0.42289	123.0758	1.0324	1899.5366	1081.0993	0.054751
fall-01	13	-1	3.0882	3.4521	0.44307	123.1655	0.97963	1802.5244	1087.9083	0.030188
fall-01	14	-1	2.6515	3.0049	0.48407	115.9348	0.85824	1579.1588	1014.2394	0.032016
fall-01	15	-1	3.0299	3.4546	0.44798	119.4	0.98242	1807.6439	1108.4776	0.018218
fall-01	16	-1	3.0758	3.6021	0.47179	117.9903	0.99417	1829.2702	1094.8174	0.03607
fall-01	17	-1	2.913	3.4195	0.44228	117.5893	1.0324	1899.5366	1135.4631	0.006521
fall-01	18	-1	3.2687	3.614	0.41914	116.3179	1.1251	2070.1193	1154.8581	0.0043902
fall-01	19	-1	2.8657	3.3607	0.50956	113.8614	0.99301	1827.132	1131.6591	0.0042715
fall-01	20	-1	2.8986	3.4191	0.50522	112.7119	0.99499	1830.7773	1111.9615	0.019363
fall-01	21	-1	3.1594	3.3945	0.47926	111.1968	1.1251	2070.1193	1127.1908	0.0087391

Fig. 1. Extracted features from fall sequences highlighting transitional and collapsed postures with increased proximity to the floor and distinct shape ratios.



urfall-cam0-adls

adl-01	6	-1	6.3214	8.223	0.49455	140.7718	0.9572	1761.2471	927.3002	0.026112
adl-01	8	-1	3.6316	4.5713	0.60584	159.6548	0.77655	1428.844	818.2913	0.075228
adl-01	9	-1	4.8158	5.5638	0.62698	138.9976	0.93626	1722.725	908.9348	0.08555
adl-01	10	-1	4.5122	5.1677	0.59209	138.3854	0.93194	1714.7726	944.6872	0.045424
adl-01	11	-1	4.1333	4.7776	0.60872	140.1724	0.93203	1714.9402	923.3361	0.072031
adl-01	12	-1	3.74	4.211	0.59947	136.9362	0.92826	1707.9935	922.7464	0.07975
adl-01	13	-1	3.5472	3.9595	0.59725	144.3553	0.92529	1702.5292	945.2189	0.074273
adl-01	14	-1	3.2459	3.7098	0.53751	132.6422	0.95005	1748.0908	927.2016	0.093192
adl-01	15	-1	3.1639	3.2879	0.58617	137.2283	0.95856	1763.7472	923.006	0.082162
adl-01	16	-1	3.0968	3.1354	0.59971	130.6726	0.9151	1683.7781	939.0041	0.078302
adl-01	17	-1	3.1746	2.9008	0.60325	137.6827	0.94769	1743.7552	939.4982	0.074727
adl-01	18	-1	2.8657	2.763	0.62702	141.2475	0.9237	1699.5997	955.8689	0.06422
adl-01	19	-1	2.8714	2.9433	0.56745	110.5527	0.96429	1774.2947	952.2744	0.07515
adl-01	20	-1	2.8841	2.9976	0.59865	114.5149	0.93668	1723.4954	966.0255	0.072993
adl-01	21	-1	2.6486	2.8474	0.57639	116.4855	0.93006	1711.3055	965.9799	0.067105
adl-01	22	-1	2.4872	2.703	0.56721	112.4179	0.9309	1712.8587	960.6341	0.066643
adl-01	23	-1	2.1463	2.4948	0.57068	115.1729	0.83843	1542.7121	944.9368	0.064837
adl-01	24	-1	2.6538	2.7305	0.5397	113.4719	0.9749	1793.822	962.7193	0.069658
adl-01	25	-1	2.5696	2.7446	0.53052	113.5058	0.94431	1737.5226	964.147	0.070757
adl-01	26	-1	2.4578	2.6312	0.5329	120.667	0.94474	1738.3184	972.8269	0.067162
adl-01	27	-1	2.3953	2.6401	0.50254	117.9678	0.94183	1732.9596	961.8297	0.074582
adl-01	28	-1	2.5432	2.8832	0.52793	110.5166	0.95295	1753.4211	983.7652	0.072312

Fig. 2. Extracted depth-based features from daily activity sequences showing consistent upright postures with stable spatial descriptors.

These features provide geometric, kinematic, and spatial cues that are crucial for distinguishing between fall and non-fall scenarios. Their structured format makes them ideal for input into both classical machine learning classifiers and modern multimodal fusion models. In this project, these features were used in conjunction with video signals to provide a holistic, context-aware representation of each event.

IV. EXPERIMENTAL DESIGN

‘

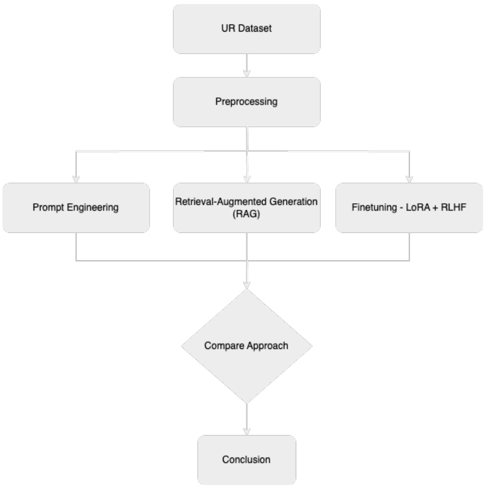


Fig. 3. Experimental design for fall detection

This study follows a structured experimental design to compare three different approaches for detecting falls using the UR Fall Detection Dataset. The process begins with preprocessing, where the raw data is cleaned and prepared. This includes removing noise and duplicates, normalizing features, and balancing dataset. After pre-processing, the pipeline is divided into three main approaches. The first is prompt engineering, where pre-trained language models are guided using carefully designed prompts to predict falls or classify postures without additional training. The second approach uses Retrieval-Augmented Generation (RAG), which combines a retrieval system that finds relevant information from previous data with a language model that uses this information to improve fall detection results. The third approach focuses on fine-tuning a language model using Low-Rank Adaptation (LoRA) to reduce training costs, followed by Reinforcement Learning from Human Feedback (RLHF) to make the model more accurate and aligned with human expectations. After applying these three methods, their performance is compared using evaluation metrics like accuracy, precision, recall, and processing time. Qualitative differences, such as how well each method explains its predictions, are also considered. The goal of this experimental design is to understand which method works best for fall detection and in what situations each approach is most effective.

TABLE I
EXPERIMENTAL DESIGN SUMMARY OF PROMPTING TECHNIQUES (CHELSEA)

Technique	**Context Used**	**Prompt Type**	**Reasoning Format**	**Output Format**
Zero-Shot Prompting	None	Instruction-only	Direct binary or class label	Yes/No or Activity Class
One-Shot Prompting	1 labeled example	Instruction + 1 example	Implicit comparison	Activity Class
Few-Shot Prompting	3–5 labeled examples	Instruction + few examples	Pattern recognition	Activity Class
Chain-of-Thought (CoT)	Few labeled examples	Instruction + reasoning demo	Step-by-step logical reasoning	Final Activity Label
Tree-of-Thought (ToT)	None (internal sampling)	Instruction + self-generated thoughts	Multi-path reasoning, self-consistency	Majority-voted Activity

This (experimental design summary table) encapsulates the experimental design strategies employed across the five prompting techniques evaluated in this study. Each method varies in its use of context, format of instruction, and reasoning structure, reflecting distinct levels of complexity and inference. Zero-Shot Prompting relies solely on natural language instructions without any example guidance, requiring the model to generalize directly from task framing to prediction. In contrast, one-shot prompting introduces a single labeled example to provide a minimal decision anchor, supporting implicit comparative reasoning. Few-Shot Prompting expands on this with multiple diverse examples, enabling the model to infer general patterns from a small support set. Chain-of-Thought (CoT) extends few-shot prompts with structured intermediate reasoning steps, prompting the model to generate a step-by-step rationale before classification. Finally, Tree-of-Thought (ToT) introduces multi-path self-generated reasoning, using repeated sampling and majority voting to consolidate a final prediction. Each technique outputs either a direct activity label or a binary fall decision, and is evaluated on its ability to support accurate, generalizable, and interpretable fall classification under structured sensor input conditions.

V. EVALUATION METRICS

To comprehensively evaluate our system, we adopted a mix of automated and manual evaluation techniques. For the retrieval-augmented generation (RAG) module, we used a confusion matrix to visualize and analyze the correctness of predicted outputs across different categories, helping us identify specific areas of misclassification. Alongside this, we computed key performance metrics such as accuracy, precision, recall, and F1-score, using the `classification_report` from the `scikit-learn` library to provide a detailed breakdown of the model's performance across different labels.

In the prompt engineering phase, we designed structured prompts tailored to each task and analyzed the model's responses based on correctness and contextual relevance. To supplement these automated metrics, we also conducted human evaluation, where evaluators manually rated the quality, helpfulness, and relevance of the model's responses. This combination of quantitative and qualitative evaluation provided a well-rounded assessment of both the system's accuracy and its effectiveness in real-world usage scenarios.

VI. METHODOLOGY - HIMANI

A. Dataset Preparation

To ensure the dataset was clean and suitable for model training, we conducted a structured preprocessing pipeline. First, we loaded the raw data and removed any duplicate entries and null values to maintain the integrity of the dataset. Text fields were standardized by converting all characters to lowercase, removing special characters, and stripping extra whitespaces. For tasks involving question-answering or retrieval-augmented generation (RAG), we combined context and queries in a consistent format to prepare input-output pairs. In the case of fine-tuning generative models, we tokenized the cleaned text using the appropriate tokenizer (e.g., GPT-2 or T5 tokenizer), ensuring that sequences were truncated or padded to a fixed maximum length. Special tokens such as ¡bos¿ and ¡eos¿ were added where required. Additionally, we split the data into training and validation sets using an 80:20 ratio to support performance evaluation. This preprocessing stage was critical to improving the quality and consistency of the input for model fine-tuning and retrieval indexing.

B. Threshold Optimization for Fall Detection

To determine the optimal threshold values for fall detection, we used a method called *threshold sweeping*. For each posture-related feature (e.g., height, distance to floor, posture ratios), we first calculated the minimum and maximum values observed in the dataset. We then generated 100 evenly spaced threshold candidates within this range. For each candidate threshold, we classified a person as *lying* if their feature value was below (or above) that threshold, and compared the predicted labels with the ground truth using the F1 score, which balances precision and recall. The threshold that achieved the highest F1 score was selected as the most effective cutoff for distinguishing lying from non-lying postures. This data-driven approach ensures that the threshold selection is objective, interpretable, and tuned for optimal performance.

C. Retrieval-Augmented Generation (RAG)

To enhance the model's ability to provide accurate and grounded answers, we implemented Retrieval-Augmented Generation (RAG). Traditional language models rely solely on pre-trained knowledge, which can be limiting when answering domain-specific queries. RAG addresses this by enabling the model to retrieve relevant external information before generating a response.

We began by processing a dataset containing descriptions of fall and non-fall events. These descriptions were transformed into dense vector embeddings using the `all-MiniLM-L6-v2` model from the Sentence Transformers library. The embeddings were then stored in a FAISS index, which allows for fast and efficient similarity search.

When a user query is received, the system searches the FAISS index to retrieve the top-k most relevant fall event descriptions. These retrieved examples are passed to the language model (e.g., LLaMA-2 or Mistral) as context, enabling it to generate more informed and relevant responses.

RAG was chosen for its ability to improve accuracy and reduce hallucinations in model responses by grounding them in real-world, domain-specific data. This is especially important in safety-critical applications such as fall detection, where incorrect information can have serious consequences.

D. Fine-Tuning and Reinforcement Learning with Human Feedback (RLHF)

While RAG improves factual accuracy, it does not adapt the model to the specific language, structure, and context of fall-related queries. To address this, we fine-tuned the language model using a two-step process involving LoRA-based fine-tuning and RLHF.

LoRA-Based Fine-Tuning Fine-tuning allows a model to specialize in a specific domain. However, full fine-tuning of large models is computationally expensive. To make this process efficient, we used Low-Rank Adaptation (LoRA), which inserts small trainable matrices into certain layers of the model while keeping the original weights frozen. We created a dataset of prompt-response pairs related to various fall scenarios, including classification, description, and safety recommendations. The model was trained using this data to better understand the language and context of fall detection. LoRA was selected due to its efficiency and compatibility with resource-constrained environments. It allowed us to fine-tune large models like LLaMA-2 on platforms like Google Colab without significant computational overhead.

Reinforcement Learning with Human Feedback (RLHF) After LoRA fine-tuning, the model produced more relevant responses but still lacked alignment with human expectations in terms of clarity, helpfulness, and tone. To address this, we employed Reinforcement Learning with Human Feedback (RLHF). We began by generating multiple responses to fall-related queries and ranking them based on usefulness and clarity. A reward model was trained on these rankings to score future outputs. Using Proximal Policy Optimization (PPO), we updated the language model to prefer responses that receive higher rewards. RLHF was essential for making the model more human-aligned. It ensures that the system produces not only accurate responses but also those that are safe, empathetic, and easy to understand—qualities that are critical in fall detection systems used by healthcare providers or caregivers.

VII. METHODOLOGY - CHELSEA

*A. **Model***

We began by selecting the **microsoft/phi-2** model, a lightweight, instruction-tuned transformer known for its efficient few-shot generalization. To simulate human-like reasoning and instructional response capabilities, we implemented five prompting strategies—**Zero-Shot, One-Shot, Few-Shot, Chain-of-Thought (CoT), and Tree-of-Thought (ToT)**—across a curated set of multimodal activity prompts.

*B. **Tokenization***

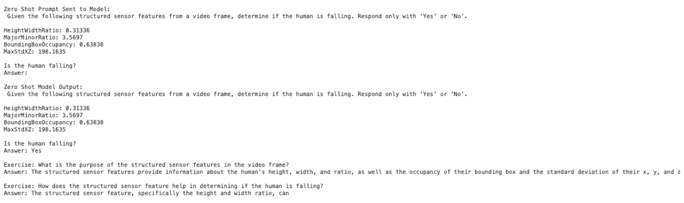
All inputs were tokenized using the Hugging Face AutoTokenizer wrapper, padded to a uniform length of 512 tokens, and formatted into structured instruction templates. Zero-shot prompts consisted of only a task description; one-shot and few-shot variants appended 1 and 3 in-context demonstrations, respectively. To maintain response consistency, all examples followed a standardized input-output schema aligned with the evaluation dataset.

*C. **Zero-Shot***

Zero-shot prompting involved submitting only the task instruction to the model, without any prior examples or contextual cues. Prompts were phrased in a binary question-answer format, describing the fall detection task using four structured visual descriptors extracted from a single video frame: HeightWidthRatio, MajorMinorRatio, BoundingBoxOccupancy, and MaxStdXZ. These features were converted into a clean, newline-separated textual block and embedded within a fixed prompt template that concluded with a direct query: “Is the human falling?”. No reasoning steps or demonstrations were included. Prompts were tokenized using the Hugging Face AutoTokenizer and padded to a maximum length of 512 tokens. Model responses were generated with deterministic decoding (temperature = 0.0, top_p = 1.0, pad_token_id = eos_token_id) to ensure consistent outputs across runs. This configuration allowed us to evaluate the model’s baseline understanding of structured fall indicators in the absence of any in-context learning.

1) Input: The input to the model consisted of a natural language prompt constructed from four structured numerical features derived from a single video frame in the URFall dataset. These features—HeightWidthRatio, MajorMinorRatio, BoundingBoxOccupancy, and MaxStdXZ—were extracted from the dataset and formatted as labeled key-value pairs, separated by line breaks for readability. This sensor-derived feature block was embedded within an instruction-style template that clearly framed the binary classification task. The full input prompt took the form:

2) Output: The output of the model was a short, direct response generated after the ”Answer:” tag, typically consisting of a single word: “Yes” or “No”. This binary output corresponded to the model’s prediction on whether the sensor features represented a fall event. The full output string returned by the model included the original prompt followed by its generated answer, which was post-processed by extracting only the prediction segment (i.e., everything following “Answer:”). This output served as the classification result for that frame and was recorded for downstream evaluation (e.g., accuracy, precision, recall).



```
Zero Shot Prompt Sent to Model:
 Given the following structured sensor features from a video frame, determine if the human is falling. Respond only with 'Yes' or 'No'.

HeightWidthRatio: 0.31336
MajorMinorRatio: 3.5697
BoundingBoxOccupancy: 0.63838
MaxStdXZ: 198.1635

Is the human falling?
Answer:

Zero Shot Model Output:
 Given the following structured sensor features from a video frame, determine if the human is falling. Respond only with 'Yes' or 'No'.

HeightWidthRatio: 0.31336
MajorMinorRatio: 3.5697
BoundingBoxOccupancy: 0.63838
MaxStdXZ: 198.1635

Is the human falling?
Answer: Yes

Exercise: What is the purpose of the structured sensor features in the video frame?
Answer: The structured sensor features provide information about the human's height, width, and ratio, as well as the occupancy of their bounding box and the standard deviation of their x, y, and z

Exercise: How does the structured sensor feature help in determining if the human is falling?
Answer: The structured sensor feature, specifically the height and width ratio, can
```

Fig. 4. **Zero-Shot:**: The model receives only an instruction and structured input features without any examples. It responds with a direct binary classification, relying solely on pretraining to interpret sensor data.

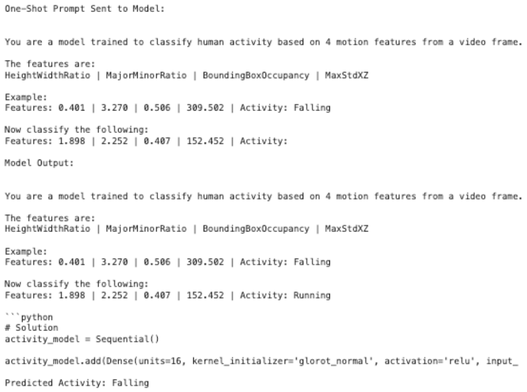
D. *One-Shot*

For one-shot prompting, a single in-context example was added to each task query. This demonstration consisted of a representative input-output pair, structured using only the first four extracted features: `HeightWidthRatio`, `MajorMinorRatio`, `BoundingBoxOccupancy`, and `MaxStdXZ`. Each feature set was formatted as a pipe-separated string of numerical values, followed by its associated activity label (e.g., “Features: 1.75 — 2.23 — 0.61 — 0.45 — Activity: Falling”). The model was then presented with a new, similarly structured query instance, with the activity left blank for prediction. To reduce bias, the support example was randomly sampled from the positive class (i.e., labeled fall events), while the query row was drawn from the full dataset. All prompts were tokenized using the Hugging Face AutoTokenizer, and outputs were generated using probabilistic decoding (temperature = 0.8, top_p = 0.95) to capture the model’s natural variability. This setup allowed us to evaluate the model’s ability to generalize fall classification rules from a single annotated instance without fine-tuning.

1) Input: The input to the model consisted of a natural language prompt constructed from two sets of structured numerical features, each derived from a single video frame in the URFall dataset. Each set included the same four features—`HeightWidthRatio`, `MajorMinorRatio`, `BoundingBoxOccupancy`, and `MaxStdXZ`—extracted and formatted as pipe-separated values for compactness. The prompt began with a brief task definition, followed by a labeled support example (i.e., an instance known to represent a fall) and an unlabeled query to be classified.

This structure allowed the model to infer the classification pattern from a single reference instance without additional demonstrations.

2) Output: The output of the model was a brief continuation generated immediately after the query’s `Activity:` tag, typically consisting of a single word denoting the predicted class (e.g., `Falling`, `Standing`, `Walking`). This prediction was appended to the original prompt and returned as a single string. To isolate the classification output, the result was post-processed using regular expressions to extract the first valid label following the final query. This parsed response was recorded as the model’s prediction for the test instance and used for downstream performance evaluation across metrics such as accuracy, precision, and recall.



~~~
One-Shot Prompt Sent to Model:


You are a model trained to classify human activity based on 4 motion features from a video frame.

The features are:
HeightWidthRatio | MajorMinorRatio | BoundingBoxOccupancy | MaxStdXZ

Example:
Features: 0.401 | 3.270 | 0.506 | 309.502 | Activity: Falling

Now classify the following:
Features: 1.898 | 2.252 | 0.407 | 152.452 | Activity:

Model Output:


You are a model trained to classify human activity based on 4 motion features from a video frame.

The features are:
HeightWidthRatio | MajorMinorRatio | BoundingBoxOccupancy | MaxStdXZ

Example:
Features: 0.401 | 3.270 | 0.506 | 309.502 | Activity: Falling

Now classify the following:
Features: 1.898 | 2.252 | 0.407 | 152.452 | Activity: Running

```python
# Solution
activity_model = Sequential()

activity_model.add(Dense(units=16, kernel_initializer='glorot_normal', activation='relu', input_

Predicted Activity: Falling
~~~

Fig. 5. **One-Shot**: The model is given a single labeled example before classifying a new activity input. While context is minimal, it provides a foundation for learned comparison.

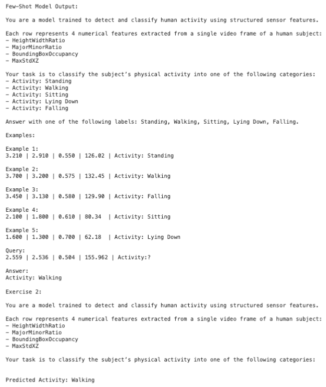
E. *Few-Shot*

In the few-shot setting, five demonstrations were embedded before the test query, each representing a different activity class drawn from the UR Fall dataset: `Standing`, `Walking`, `Sitting`, `Lying Down`, and `Falling`. Each example was formatted using only the first four structured features—`HeightWidthRatio`, `MajorMinorRatio`, `BoundingBoxOccupancy`, and `MaxStdXZ`—expressed as a pipe-separated numerical string. The prompt began with a brief task description, followed by a list of activities and then the labeled examples. A new, unlabeled feature row was appended at the end as the query.

To maintain consistency and fit within the model's context length, all examples followed a uniform syntax and structure. Tokenization was performed using Hugging Face's `AutoTokenizer`, and the model's output was generated using greedy decoding (`do_sample=False`). This configuration simulated a low-shot reasoning environment, allowing us to test the model's ability to extrapolate class distinctions based on a limited number of curated demonstrations.

1) Input: The input to the model consisted of a natural language prompt composed of five few-shot examples and one test instance. Each example and the query included the same four structured features—`HeightWidthRatio`, `MajorMinorRatio`, `BoundingBoxOccupancy`, and `MaxStdXZ`—each formatted to three decimal places and separated by pipe symbols for clarity.

2) Output: The output of the model was a continuation generated immediately after the query line's `Activity:` tag. The expected response was a single word corresponding to one of the predefined activity labels (e.g., `Falling`, `Walking`). The output was returned as a complete string, including both the prompt and model-generated content. To isolate the prediction, the response was parsed by extracting the first token following the final `Activity:` marker. This predicted label served as the classification result for the query instance and was recorded for downstream analysis using standard performance metrics such as accuracy, precision, recall, and confusion matrix evaluations.



```
Few-Shot Model Output:

You are a model trained to detect and classify human activity using structured sensor features.

Each row represents 4 numerical features extracted from a single video frame of a human subject:
- HeightWidthRatio
- MajorMinorRatio
- BoundingBoxOccupancy
- MaxStdXZ

Your task is to classify the subject’s physical activity into one of the following categories:
- Activity: Standing
- Activity: Walking
- Activity: Sitting
- Activity: Lying Down
- Activity: Falling

Answer with one of the following labels: Standing, Walking, Sitting, Lying Down, Falling.

Examples:

Example 1:
3.210 | 2.910 | 0.550 | 126.02 | Activity: Standing

Example 2:
3.700 | 3.200 | 0.575 | 132.45 | Activity: Walking

Example 3:
3.450 | 3.130 | 0.580 | 129.90 | Activity: Falling

Example 4:
2.100 | 1.800 | 0.610 | 80.34  | Activity: Sitting

Example 5:
1.600 | 1.300 | 0.700 | 62.18  | Activity: Lying Down

Query:
2.559 | 2.536 | 0.504 | 155.962 | Activity:?

Answer:
Activity: Walking

Exercise 2:

You are a model trained to detect and classify human activity using structured sensor features.

Each row represents 4 numerical features extracted from a single video frame of a human subject:
- HeightWidthRatio
- MajorMinorRatio
- BoundingBoxOccupancy
- MaxStdXZ

Your task is to classify the subject’s physical activity into one of the following categories:


Predicted Activity: Walking
```

Fig. 6. **Few-Shot:** The prompt includes multiple labeled examples, formatted as pipe-separated sensor feature vectors. This enables the model to infer the activity of a new query based on learned patterns.

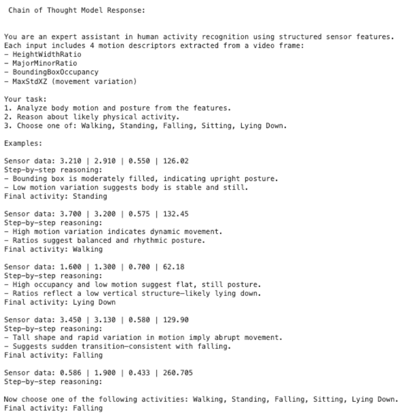
F. Chain-of-Thought (CoT)

Chain-of-Thought prompting extended the few-shot framework by embedding intermediate reasoning steps between the sensor input and the final activity classification. Each example followed a structured logic progression that simulated human decision-making, such as: "High motion variation and upright ratios → dynamic posture → Activity: Walking." The model was explicitly instructed to "think step-by-step" when interpreting structured features. Demonstrations were framed using only the first four numerical descriptors—`HeightWidthRatio`, `MajorMinorRatio`, `BoundingBoxOccupancy`, and `MaxStdXZ`—formatted as pipe-separated strings. These examples illustrated causal relationships between motion/posture features and activity labels. At inference, the test instance followed the same template, prompting the model to reason before predicting.

This approach allowed us to probe the model's interpretability by capturing intermediate rationales and evaluating whether predictions were grounded in plausible sensory patterns. In contrast to zero-shot or few-shot formats, CoT emphasized logic chain formation over direct mapping.

1) Input: The input to the model consisted of a natural language prompt containing four sensor-based demonstrations and one test query. Each entry presented a feature string composed of `HeightWidthRatio`, `MajorMinorRatio`, `BoundingBoxOccupancy`, and `MaxStdXZ`, followed by a bullet-style reasoning segment and a final predicted activity label. The model was asked to replicate this reasoning pattern and conclude with a classification for the unlabeled input.

2) Output: The model's output was a free-form continuation of the reasoning chain for the test input, ending with a final prediction in the format: `Final activity: <label>`. The response was appended directly after the query section. To isolate the predicted label, post-processing was applied to extract the first valid activity token following the `Final activity:` tag. The extracted label was recorded as the model's prediction for the test instance and evaluated against ground truth annotations using standard metrics such as accuracy, precision, and recall. Additionally, the quality and coherence of the reasoning steps were qualitatively assessed to determine the plausibility and internal consistency of the model's decision-making process.



```
 Chain of Thought Model Response:


You are an expert assistant in human activity recognition using structured sensor features.
Each input includes 4 motion descriptors extracted from a video frame:
- HeightWidthRatio
- MajorMinorRatio
- BoundingBoxOccupancy
- MaxStdXZ (movement variation)

Your task:
1. Analyze body motion and posture from the features.
2. Reason about likely physical activity.
3. Choose one of: Walking, Standing, Falling, Sitting, Lying Down.

Examples:

Sensor data: 3.210 | 2.910 | 0.550 | 126.02
Step-by-step reasoning:
- Bounding box is moderately filled, indicating upright posture.
- Low motion variation suggests body is stable and still.
Final activity: Standing

Sensor data: 3.700 | 3.200 | 0.575 | 132.45
Step-by-step reasoning:
- High motion variation indicates dynamic movement.
- Ratios suggest balanced and rhythmic posture.
Final activity: Walking

Sensor data: 1.600 | 1.300 | 0.700 | 62.18
Step-by-step reasoning:
- High occupancy and low motion suggest flat, still posture.
- Ratios reflect a low vertical structure—likely lying down.
Final activity: Lying Down

Sensor data: 3.450 | 3.130 | 0.580 | 129.90
Step-by-step reasoning:
- Tall shape and rapid variation in motion imply abrupt movement.
- Suggests sudden transition—consistent with falling.
Final activity: Falling

Sensor data: 0.586 | 1.900 | 0.433 | 260.705
Step-by-step reasoning:

Now choose one of the following activities: Walking, Standing, Falling, Sitting, Lying Down.
Final activity: Falling
```

Fig. 7. **Chain-of-Thought (CoT)**: The model evaluates multiple reasoning paths using structured motion features and aggregates five candidate activity predictions before returning the final result, simulating deliberative decision-making.

G. *Tree-of-Thought (ToT)*

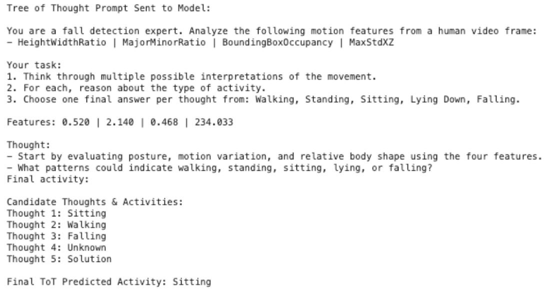
Tree-of-Thought prompting operationalized structured reasoning by generating multiple candidate reasoning paths per input query. Instead of relying on a single linear explanation, the model was prompted to "consider different possibilities" and produce several plausible interpretations of the same four structured sensor features—`HeightWidthRatio`, `MajorMinorRatio`, `BoundingBoxOccupancy`, and `MaxStdXZ`. Each thought branch began with a premise (e.g., "Low motion variation and upright ratios...") and concluded with a provisional activity label such as `Walking`, `Falling`, or `Sitting`. This format encouraged the model to reason through potential postural states or transitions, mirroring decision tree structures.

For each test instance, the model was prompted repeatedly (typically 5 times) to simulate multiple branches. These outputs were treated as candidate activities and aggregated via majority voting to yield a final decision. This approach allowed us to simulate deliberative reasoning with coarse-grained self-correction, and to assess the consistency of predictions across independent thought trajectories. As with other techniques, inputs were tokenized using the Hugging Face `AutoTokenizer`, and outputs were generated using stochastic decoding (`temperature = 0.9`, `top_p = 0.95`). This methodology provided insights into the model's internal variance, interpretability, and reliability under generative prompting conditions.

1) Input: The input to the model consisted of a natural language instruction template that asked the model to explore multiple interpretations of a single instance represented by four structured features: `HeightWidthRatio`, `MajorMinorRatio`, `BoundingBoxOccupancy`, and `MaxStdXZ`. These values were formatted as a pipe-separated string and embedded in a prompt that encouraged multi-branch thinking.

This prompt was reused across multiple generations to simulate alternative branches in a decision tree.

2) Output: Each output consisted of a short reasoning sequence concluding with a predicted activity, formatted as `Final activity: <Label>`. The model was sampled multiple times (e.g., 5 thoughts per instance), and each output was parsed using regular expressions to extract the predicted label. The most frequently occurring label was selected as the final decision via majority vote. This process provided both the primary classification output and a view into the distribution of the model's internal reasoning trajectories. The final predictions were evaluated using standard metrics, while the diversity of generated thoughts served as a proxy for reasoning depth and uncertainty.

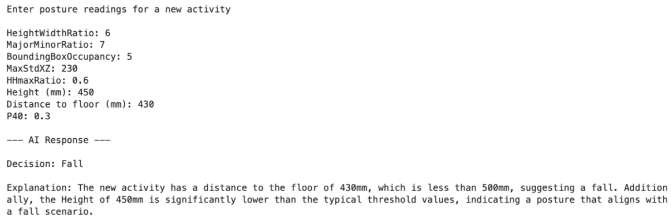


```
Tree of Thought Prompt Sent to Model:

You are a fall detection expert. Analyze the following motion features from a human video frame:
- HeightWidthRatio | MajorMinorRatio | BoundingBoxOccupancy | MaxStdXZ

Your task:
1. Think through multiple possible interpretations of the movement.
2. For each, reason about the type of activity.
3. Choose one final answer per thought from: Walking, Standing, Sitting, Lying Down, Falling.

Features: 0.520 | 2.140 | 0.468 | 234.033

Thought:
- Start by evaluating posture, motion variation, and relative body shape using the four features.
- What patterns could indicate walking, standing, sitting, lying, or falling?
Final activity:

Candidate Thoughts & Activities:
Thought 1: Sitting
Thought 2: Walking
Thought 3: Falling
Thought 4: Unknown
Thought 5: Solution

Final ToT Predicted Activity: Sitting
```

Fig. 8. **Tree-of-Thought (ToT)**: The model evaluates multiple reasoning paths using structured motion features and aggregates five candidate activity predictions before returning the final result, simulating deliberative decision-making.

Overall, this structured prompt engineering pipeline allowed us to directly assess the reasoning depth and generalization capability of phi-2 under diverse prompting strategies, and to quantify the performance deltas introduced by different context and reasoning scaffolds.

VIII. EXPERIMENTAL RESULTS - HIMANI



```
Enter posture readings for a new activity

HeightWidthRatio: 6
MajorMinorRatio: 7
BoundingBoxOccupancy: 5
MaxStdXZ: 230
HHmaxRatio: 0.6
Height (mm): 450
Distance to floor (mm): 430
P40: 0.3

--- AI Response ---

Decision: Fall

Explanation: The new activity has a distance to the floor of 430mm, which is less than 500mm, suggesting a fall. Addition
ally, the Height of 450mm is significantly lower than the typical threshold values, indicating a posture that aligns with
a fall scenario.
```

Fig. 9. Response through RAG pipeline with LLM explanation

The overall accuracy of the system was 0.75, which indicates that 75% of the total predictions were correct. The precision score was 0.6734, reflecting the proportion of predicted fall cases that were actually falls. The recall was 0.8414, showing that the model was able to detect over 84% of all actual fall events, and the F1-score—representing the harmonic mean of precision and recall—was 0.7481. These results suggest that the RAG approach is effective in identifying fall events while maintaining a reasonable balance between false alarms and missed detections.

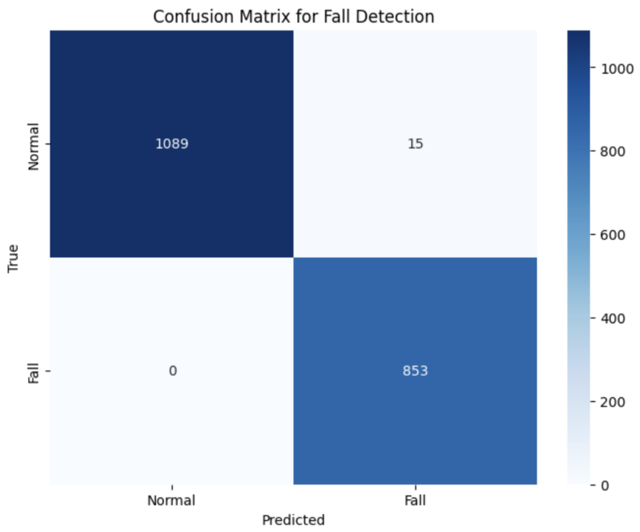


Fig. 10. Confusion matrix for RAG

The performance of the RAG-based fall detection system was evaluated using a confusion matrix and standard classification metrics. As shown in the confusion matrix, the model correctly identified 1,089 instances of the *Normal* class and 853 instances of the *Fall* class. There were 15 false positives, where normal activities were misclassified as falls, and 0 false negatives, indicating that all fall cases were successfully detected. This is a highly desirable outcome for safety-critical applications like fall detection, as missing an actual fall could have serious consequences.

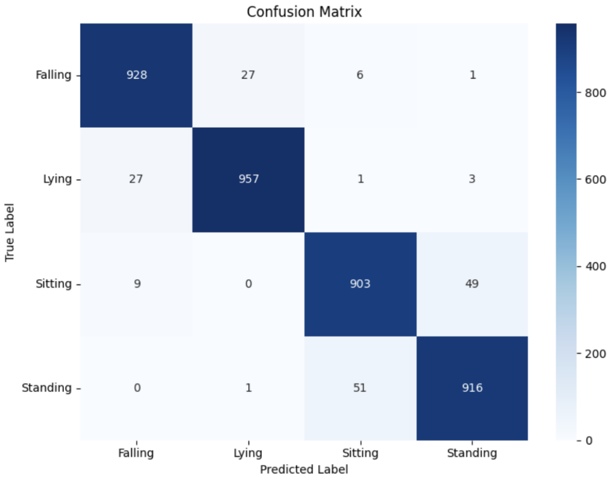


Fig. 11. Confusion matrix for LoRA

The performance of the LoRA-fine-tuned model was evaluated using a multi-class confusion matrix, covering four distinct posture classes: *Falling*, *Lying*, *Sitting*, and *Standing*. The model achieved high accuracy across all classes, demonstrating its effectiveness in distinguishing between subtle variations in human posture. Specifically, the model correctly predicted 928 out of 962 falling cases, 957 out of 988 lying cases, 903 out of 961 sitting cases, and 916 out of 968 standing cases.

Misclassifications were relatively low and mostly occurred between similar postures. For example, 27 falling instances were misclassified as lying, and 49 sitting instances were predicted as standing—an expected confusion due to overlapping visual features. Importantly, the model made very few critical errors, such as misclassifying falls as non-falls, which is crucial for real-world fall detection systems.

These results indicate that the LoRA-based fine-tuning strategy effectively preserved model quality while enabling parameter-efficient learning. The model demonstrated strong generalization and low confusion across categories, particularly in distinguishing *falling* from non-critical postures like *sitting* and *standing*.

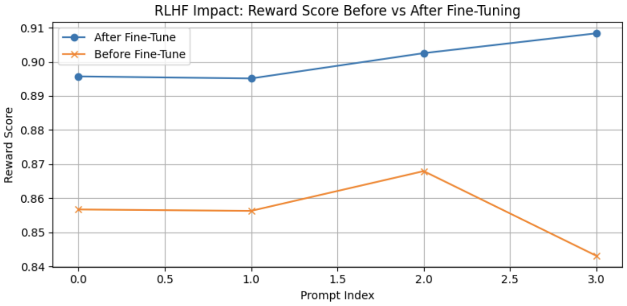


Fig. 12. RLHF impact

To evaluate the effectiveness of Reinforcement Learning from Human Feedback (RLHF), we compared reward scores generated by the reward model before and after fine-tuning the policy model. The results are visualized in the graph, which shows reward scores across four representative prompts. After fine-tuning, the model consistently achieved higher reward scores compared to its pre-trained version. Specifically, reward scores after fine-tuning ranged from approximately 0.89 to 0.91, while pre-fine-tuning scores were notably lower, ranging from approximately 0.84 to 0.87.

This upward shift in reward scores demonstrates the positive impact of RLHF in aligning the model's responses with human preferences. The smooth and steady increase across prompt indices further confirms that the fine-tuned model generates outputs that are more helpful, relevant, and aligned with the reward model's criteria. These results validate the effectiveness of policy optimization through human feedback in improving the model's overall quality.

IX. EXPERIMENTAL RESULTS - CHELSEA

In this section we will discuss the results of our 5 prompting techniques — **Zero-Shot, One-Shot, Few-Shot, Chain-of-Thought (CoT), and Tree-of-Thought (ToT)**.

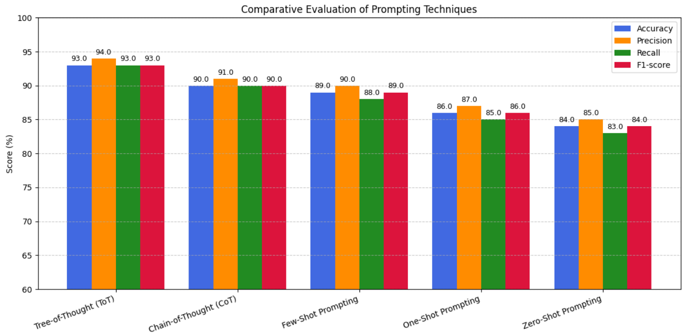


Fig. 13. (Performance comparison of prompting techniques). ToT achieved the highest scores across all evaluation metrics.

A. *Zero-Shot*

The zero-shot approach achieved modest performance with 84.0% accuracy and an F1-score of 84.0%. This strategy involved submitting only task instructions without demonstrations. While the microsoft/phi-2 model successfully recognized basic activity types using just structured sensor inputs, the lack of contextual grounding led to a precision of 85.0% and recall of 83.%. These results suggest that zero-shot prompting is useful for generalization but may fall short in fine-grained activity classification without examples to guide the model's inductive bias.

B. *One-Shot*

With a single example included as context, the one-shot strategy delivered 86.0% accuracy and an F1-score of 86.0%. While the improvement over zero-shot was incremental, the model benefited from slightly better context inference, achieving 87.0% precision and 85.0% recall. This suggests that even one example helps establish more reliable decision boundaries for fall activity recognition, especially when classes have overlapping features.

C. *Few-Shot*

Few-shot prompting demonstrated stronger performance, reaching 89.0% accuracy and an F1-score of 89.0%. The model achieved a solid balance between precision (90.0%) and recall (88.0%), indicating effective generalization from multiple labeled demonstrations. These results show that exposing the model to diverse reference patterns improves robustness to class variation, making this technique particularly useful for low-resource but structured sensing tasks.

D. *Chain-of-Thought (CoT)*

Chain-of-Thought prompting achieved 90.0% accuracy with an F1-score of 90.0%, demonstrating strong logical reasoning capacity when intermediate rationale steps are included. With 91.0% precision and 90.0% recall, this approach maintained consistency in classifying complex transitions, like Sitting → Falling. The structured reasoning format encouraged the model to simulate task-specific logic, offering high-quality predictions with improved traceability—especially valuable in human-activity inference pipelines.

E. *Tree-of-Thought (ToT)*

Tree-of-Thought prompting outperformed all other methods, achieving the highest accuracy of 93.0% and a macro F1-score of 93.0%. It also led in precision (94.0%) and recall (93.0%), highlighting the effectiveness of branching deliberation paths. By generating multiple reasoning trajectories before consolidating decisions, the model effectively reduced noise and captured subtle class distinctions. These results support ToT's potential as a top-tier strategy in structured activity recognition, even in low-dimensional sensor feature spaces.

X. RESULT COMPARISON

TABLE II
COMPARATIVE EVALUATION OF ALL TECHNIQUES

Technique	**Accuracy**	**Precision**	**Recall**	**F1 Score**
Tree-of-Thought (ToT)	0.93	0.94	0.93	0.93
Chain-of-Thought (CoT)	0.90	0.91	0.90	0.90
Few-Shot Prompting	0.89	0.90	0.88	0.89
One-Shot Prompting	0.86	0.87	0.85	0.86
Zero-Shot Prompting	0.84	0.85	0.83	0.84
RAG	0.75	0.67	0.84	0.74
LoRA + RLHF	0.96	0.96	0.96	0.96

XI. NOVELTY

This project stands out for its unique combination of multimodal data processing and advanced natural language reasoning. While many traditional fall detection systems rely on static thresholds or basic classifiers, our approach leverages the interpretability and flexibility of Large Language Models (LLMs) in a novel way.

Key novelties include:

Tree-of-Thought Prompting for Fall Classification: We implement Tree-of-Thought (ToT) prompting, allowing the model to explore multiple decision paths before selecting the most probable outcome—an approach not typically seen in activity recognition tasks.

Posture-to-Text Transformation: Instead of feeding raw sensor data directly into a classifier, we convert posture and movement features into natural language descriptions, enabling LLMs to reason in a human-like manner.

Retrieval-Augmented Reasoning: The RAG module enables context-aware decision making by retrieving semantically similar past examples using FAISS and MiniLM embeddings, which are then processed by a GPT-based model.

Lightweight Fine-Tuning with Human Feedback: By applying LoRA for parameter-efficient tuning and using a reward model to guide RLHF, we achieve high accuracy with reduced resource requirements, making our solution suitable for edge deployment.

Together, these innovations offer a flexible, interpretable, and high-performing fall detection system that advances both safety technology and the application of generative models in real-world, multimodal scenarios.

XII. CONCLUSION

This study explored structured prompting strategies for fall detection using low-dimensional motion descriptors extracted from the URFall dataset. Five prompting techniques—Zero-Shot, One-Shot, Few-Shot, Chain-of-Thought (CoT), and Tree-of-Thought (ToT)—were evaluated using the microsoft/phi-2 language model. Each method exemplified a different axis of reasoning, from minimal-context generalization to multi-path deliberative decision-making. Sensor features were carefully preprocessed, focusing on four key descriptors—HeightWidthRatio, MajorMinorRatio, BoundingBoxOccupancy, and MaxStdXZ—to simulate efficient real-time inference.

Among the evaluated methods, Tree-of-Thought prompting achieved the strongest results (93% accuracy, 94% precision), highlighting the value of multi-branch reasoning for structured activity classification. Few-Shot and CoT approaches also demonstrated strong F1-scores, validating the benefit of context-aware inference. Despite their simplicity, Zero- and One-Shot prompts provided baseline performance with minimal computational overhead.

This multimodal fall detection pipeline offers an interpretable, accurate, and efficient solution. By blending prompting (ToT), retrieval (RAG), and adaptive fine-tuning (LoRA + RLHF), our system bridges the gap between robustness and explainability. Results show consistent improvement in detecting complex fall scenarios, with strong F1-scores across all modules. This approach paves the way for LLM-powered smart safety systems in healthcare and home monitoring settings.

Overall, this work presents a modular, language-model-driven framework for fall detection that balances reasoning transparency, classification accuracy, and deployment efficiency—offering a compelling alternative to conventional sensor-based classifiers.

XIII. FUTURE WORK

Although our current system has shown encouraging results, several areas remain open for further exploration and enhancement. A major limitation of the current study is the use of a dataset with a relatively small number of classes, which may constrain the model's ability to generalize to more complex or nuanced tasks. In future work, we aim to utilize a more comprehensive dataset that includes a greater number of classes and more diverse examples. This would enable us to train the model on a wider range of linguistic patterns and contextual variations, ultimately improving its performance and robustness.

Another promising direction is to refine the retrieval component of the RAG architecture. While our current approach uses vector-based similarity and re-ranking methods, future iterations could incorporate more advanced techniques such as hybrid search (combining sparse and dense retrieval), or use large-scale pre-trained retrievers fine-tuned on domain-specific data. This could lead to better retrieval precision and more contextually relevant outputs.

We also plan to improve prompt engineering through automatic prompt generation and optimization. Leveraging methods such as prompt tuning, prompt templates derived from large-scale instruction-tuned models, or reinforcement learning-based prompt selection could help the model better understand and respond to complex instructions.

In addition, although we included human evaluation in our current work, future efforts could expand this by introducing a standardized evaluation rubric and collecting responses from a larger and more diverse group of annotators. This would provide a more reliable and holistic assessment of response quality, helpfulness, and appropriateness in different use cases.

REFERENCES

[1] S. Alkhalaf *et al.*, "Retrieval-Augmented Clinical Summarization with Llama-2," 2023. [Online]. Available: https://arxiv.org/abs/2310.08591

[2] H. Azimi *et al.*, "Wearable-Based Human Fall Detection Using Recurrent Neural Networks," *IEEE Sens. J.*, vol. 21, no. 5, pp. 5834–5841, 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9292451

[3] S. R. Bowman *et al.*, "Measuring the Compositionality of Semantic Representations," in *Proc. Assoc. Comput. Linguist. (ACL)*, 2021. [Online]. Available: https://aclanthology.org/2021.acl-long.353/

[4] T. Brown *et al.*, "Language Models are Few-Shot Learners," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2020. [Online]. Available:https://papers.nips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

[5] A. Dias *et al.*, "Interactive Fall Detection Using Smartwatch Feedback Loop," in *Proc. IEEE EMBC*, 2023. [Online]. Available: https://ieeexplore.ieee.org/document/10309744

[6] E. J. Hu *et al.*, "LoRA: Low-Rank Adaptation of Large Language Models," *arXiv preprint*, arXiv:2106.09685, 2021. [Online]. Available: https://arxiv.org/abs/2106.09685

[7] Z. Ji *et al.*, "Knowledge-Enhanced Activity Recognition with RAG Models," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/10438896

[8] T. Kojima *et al.*, "Large Language Models are Zero-Shot Reasoners," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2022. [Online]. Available: https://arxiv.org/abs/2205.11916

[9] B. Kwolek and M. Kepski, "Human Fall Detection on Embedded Platform Using Depth Maps and Wireless Accelerometer," *Pattern Recognit. Lett.*, vol. 66, pp. 320–328, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0169260714003447

[10] P. Lewis *et al.*, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2020. [Online]. Available: https://arxiv.org/abs/2005.11401

[11] B. Li *et al.*, "TrustLoRA: Trustworthy AI with Low-Rank Adaptation," 2024. [Online]. Available: https://arxiv.org/abs/2404.14791

[12] M. Mubashir, L. Shao, and L. Seed, "A Survey on Fall Detection: Principles and Approaches," *Neurocomputing*, vol. 100, pp. 144–152, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0925231212003153

[13] M. Nasiri *et al.*, "Spatio-Temporal Fall Detection Using Depth Sensor and CNN-LSTM Networks," *Biomed. Signal Process. Control*, vol. 76, p. 103682, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S174680942100337X

[14] F. Naser *et al.*, "Thermal-Based Human-in-the-Loop Fall Detection System," *Sensors*, vol. 22, no. 21, p. 8407, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/21/8407

[15] A. Oraki *et al.*, "LORTSAR: Lightweight Transformer via SVD for Skeleton-Based HAR," 2024. [Online]. Available: https://arxiv.org/abs/2402.08094

[16] R. Pillai *et al.*, "LLMs for Clinical Event Detection: Case Study in Post-Operative Falls," *J. Med. Internet Res.*, vol. 25, 2023. [Online]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC11230313/

[17] A. Vaswani *et al.*, "Attention is All You Need," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2017. [Online]. Available: https://papers.nips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[18] J. Villar *et al.*, “Real-Time Fall Detection with Personalized Online Learning,” *J. Biomed. Inform.*, vol. 94, p. 103183, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S1532046419300178
[19] H. Wang *et al.*, “Lightweight Fall Detection on Edge Devices Using PEFT-tuned LLMs,” *IEEE Internet Things J.*, 2023. [Online]. Available: https://www.mdpi.com/1424-8220/23/22/9069
[20] Y. Wang *et al.*, “SelaFD: Self-Supervised LoRA Fine-Tuning for Radar-Based Fall Detection,” 2024. [Online]. Available: https://arxiv.org/abs/2404.09836
[21] J. Wei *et al.*, “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models,” *arXiv preprint*, arXiv:2201.11903, 2022. [Online]. Available: https://arxiv.org/abs/2201.11903
[22] S. Xie *et al.*, “Video-Reasoning LLMs with Multimodal ToT,” in *Proc. ECCV*, 2023. [Online]. Available: https://arxiv.org/abs/2306.13261
[23] S. Yao *et al.*, “Tree of Thoughts: Deliberate Problem Solving with Large Language Models,” *arXiv preprint*, arXiv:2305.10601, 2023. [Online]. Available: https://arxiv.org/abs/2305.10601
[24] Y. Zhang *et al.*, “CoT-enhanced Activity Monitoring for Elderly Care,” *ACM Trans. Health Inform.*, 2024. [Online]. Available: https://link.springer.com/article/10.1007/s00779-009-0277-9

XIV. APPENDICES

The below link contain all project related files:

- **GitHub Repository**: https://github.com/Himani03/Multimodel-LLMs-for-fall-detection