WiSentinel: A Novel Approach to Human Detection and Pose Classification Using Wi-Fi Signals

Ilya Nyrkov
Bao Lam Bui
Shivam Singh Chauhan
Max Hufnagel
Muhammad Amir Rafiq Mohd Rafee

23.07.2025

Abstract

This report details the design, implementation, and evaluation of WiSentinel, a novel cyber-physical system for human detection and pose classification. Unlike conventional methods that rely on visual cameras or wearable devices, WiSentinel leverages passive Wi-Fi signals and Channel State Information (CSI) to identify and classify human presence and posture in a privacy-preserving manner. A significant challenge addressed by this project was the (almost) complete absence of pre-existing, ready-to-use toolchains for our specific application. Consequently, almost all hardware configurations, data collection protocols, signal processing pipelines, and machine learning models were developed from scratch by our team. This report outlines our methodology for the creation, training of our tools for the final purpose of pose estimation. We also discuss the significant problems we faces, mainly data scarcity and issues regarding lack of prior work done in the field, and propose future improvements.

1 Introduction

The increasing demand for intelligent systems capable of perceiving their environment without compromising privacy and making people uncomfortable has led to a growing interest in alternative sensing technologies. Traditional surveillance methods, such as cameras and wearable sensors, are often intrusive and raise privacy concerns. Our project, WiSentinel, explores a fundamentally different design approach: non-intrusive human sensing using commonly used Wi-Fi signals. By leveraging Channel State Information (CSI), a very useful metric that describes how a Wi-Fi signal is affected by its environment, we can detect and classify human activity through walls and even in no-light conditions (unlike traditional cameras).

The central premise of WiSentinel is that the human body's movement and posture create unique, time-varying changes in a Wi-Fi signal's amplitude and phase. Our system captures these minute changes and processes them using a custom-built machine learning pipeline in order to infer human presence and specific poses.

A defining characteristic of this project was the absence of a pre-existing, easily modifiable toolset. While already existing literature provided useful, theoretical foundations, the practical implementation had to be done basically from scratch. This included router firmware modifications, the data sanitization and resulting pipeline, as well as 3 machine learning models (human presence, pose estimation, position estimation). Every component of the system was developed from the ground up to meet our project requirements. This paper documents our methodology, from the initial hardware setup to the final, functional prototype.

2 System Architecture

The WiSentinel system is a modular, end-to-end solution composed of several interconnected components, as depicted in Figure 1. This architecture ensures that each part can be developed, modified and improved independently.

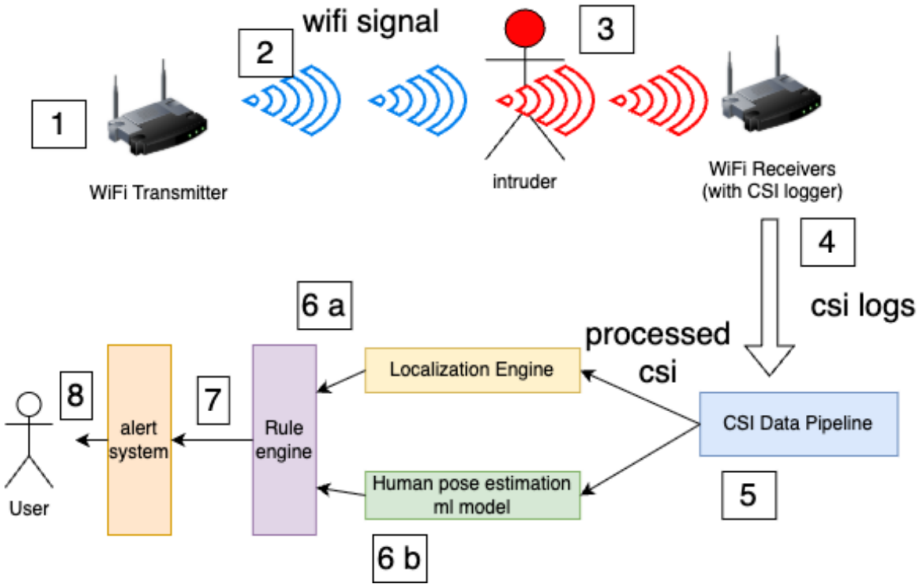

Figure 1: The demonstrative workflow of the WiSentinel system, from signal transmission to user alert.

1. **Initialization**: System armed for monitoring.
2. **Transmission**: WiFi transmitter continuously broadcasts WiFi waves into space.
3. **Reflection**: Person disturbs signal pattern causing several effects like absorption, reflection, scattering, attenuation. The patterns of these signal changes can be used to build a mapping to human poses.
4. **Capture**: CSI logs generated and streamed for processing.
5. **Processing**: CSI preprocessed into usable format (sanitization, feature calculation).
6. **Inference**:
   a. Location estimated: $x = 100, y = 150$
   b. Pose classified: crouching
7. **Evaluation**: Rule engine matches conditions:

   "time == '22:00' and pose == 'crouching' and location == (100,150)"
8. **Notification**: Alert sent to an app on user's phone (or any other configured app) with estimated pose and location.

2.1 Hardware Setup

The physical foundation of our system consists of two TP-Link Archer C7 routers (v2), configured as a Wi-Fi transmitter and a CSI receiver. These routers were specifically chosen for their compatibility with custom firmware that allows access to the normally

inaccessible CSI data. The transmitter continuously broadcasts Wi-Fi signals that are affected by the environment and any human presence. The receiver, equipped with the necessary firmware, captures these altered signals and logs the CSI data for live processing. The process of getting this setup working was a significant challenge, as commercial router companies do not provide official tools or documentation for low-level modifications. CSI data is basically a raw wi-fi signal so often, vendors also directly restrict low-level meddling with their systems, which is why many routers will not work. In addition, no proper official guidelines were available, resulting in a substantial amount of work to be done from scratch, including flashing custom OpenWrt firmware and patching the ath9k driver to enable CSI capture.

2.2 Software and Toolchain

The core of our system's functionality is driven by a custom software stack built entirely in Python. Except for external recvCSI utility which is open source and written in C to make it as small as possible for a reduced router's disk storage.

- **Router Firmware:** The routers were flashed with a custom-built OpenWrt firmware. A key step was patching the `ath9k` driver to enable low-level CSI capture.
- **CSI Data Capturing:** The Atheros CSI Tool (utility written in C) was used to capture raw Wi-Fi packets, which are then saved in a `.dat` format.
- **Codebase:** The entire processing and modeling pipeline was developed using Python libraries. Key components include:
  - `csiread`: Custom library for parsing the raw `.dat` files.
  - `dataSanitization`: Custom library for raw data sanitization and preparation.
  - `NumPy`, `Pandas` and `Matplotlib`: Used for numerical operations and data visualization.
  - `Tensorflow`: Used for machine learning part of a project.
  - `FastAPI`: Utilized to create a communication backend that connects the processing pipeline to the alerting system.

3 Data Collection and Calibration

An important phase of the project was the creation of a high-quality, labeled dataset. The absence of a suitable pre-existing dataset for our specific poses and environment meant this task was more time consuming then initially assumed. There are other similar datasets but after we trained a model on them, inference on data in our room showed terrible results. We stuck with recording and labeling our own data. Our data collection process improved over the course of the project to overcome significant hurdles. Our initial attempt to record data in the university library proved unsuccessful due to substantial ambient noise (many people in close proximity) and lack of quiet facilities. This severely compromised the quality of the dataset showed us the importance of a controlled, low-noise recording environment.

In order to address to these issues, we made the decision to completely re-record our dataset in a team member's apartment. This second effort of data collection resulted in a

dataset of far superior quality, characterized by a significantly lower signal-to-noise ratio and a consistent environment with fewer disturbances. For the final version of WiSentinel, we exclusively used this second dataset to train and validate our models. While the project functions effectively as a proof-of-concept, the limited time and manpower available to our team meant we were unable to collect the hundreds of hours of data typically used in similar research projects. Had we been afforded a normal amount of data and resources, the performance and accuracy of our models would have been substantially enhanced. Despite these resource constraints, the successful completion of the project confirms the validity and effectiveness of our methodology and tools.

It is possible to improve this setup even further. An automated setup can be used: kinect camera and running pose estimation model from images. There are much more pose estimation models from video. They can be used to automatically label CSI data by comparing two streams of data: image and csi.

3.1 Recording Setup

Our final data collection was performed in a dedicated room. The setup involved a fixed-position transmitter and receiver, which defined a 3x3 grid within a rectangular "sensing area" for a person to perform poses (Figure 2).

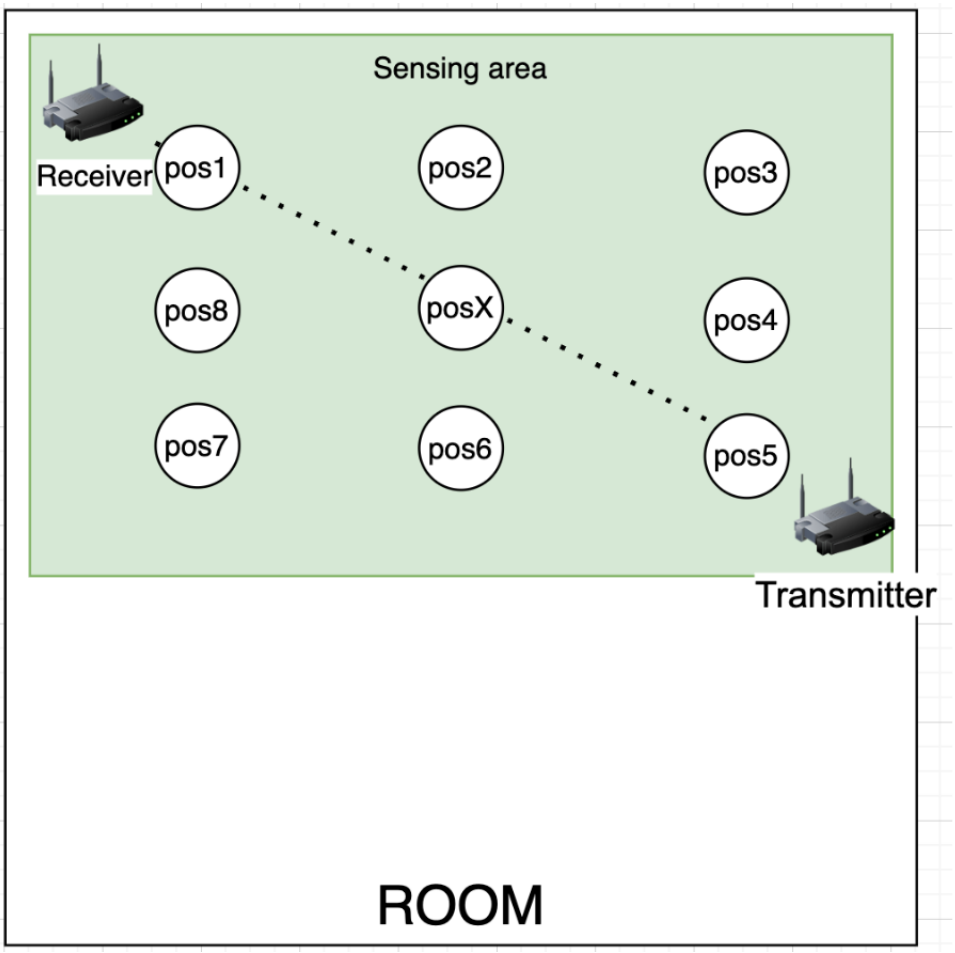

Figure 2: The data recording setup, showing the transmitter, receiver, and the 3x3 sensing grid.

The recording process involved recording a person performing specific poses (`standing`, `crouching`) at each of the nine grid positions. To improve the model's robustness, each pose was also recorded in four different body orientations relative to the Wi-Fi link: 1) facing transmitting router, 2) facing receiving router, 3,4) facing perpendicular line to a diagonal between receiving and transmitting routers. This systematic approach was essential for gathering a diverse dataset that accounts for signal variations due to position and orientation.

4 Methodology

Our methodology represents a novel, end-to-end pipeline that transforms raw CSI data into actionable pose and presence classifications. This process is divided into three main stages: Data sanitization, feature extraction (calculation), and machine learning model training. For real application instead machine learning model training we run inference for received data. The entire pipeline was custom-designed for this project, as no pre-existing solutions were suitable for our requirements. Our pipeline was designed to work with live data as well i.e. continuous stream of CSI tensors.

4.1 CSI Data Processing Pipeline

Raw data from the Atheros CSI Tool is highly susceptible to noise and hardware artifacts that must be corrected before model training. Our custom CSI Sanitization component performs the following steps, as shown in Figure 3:

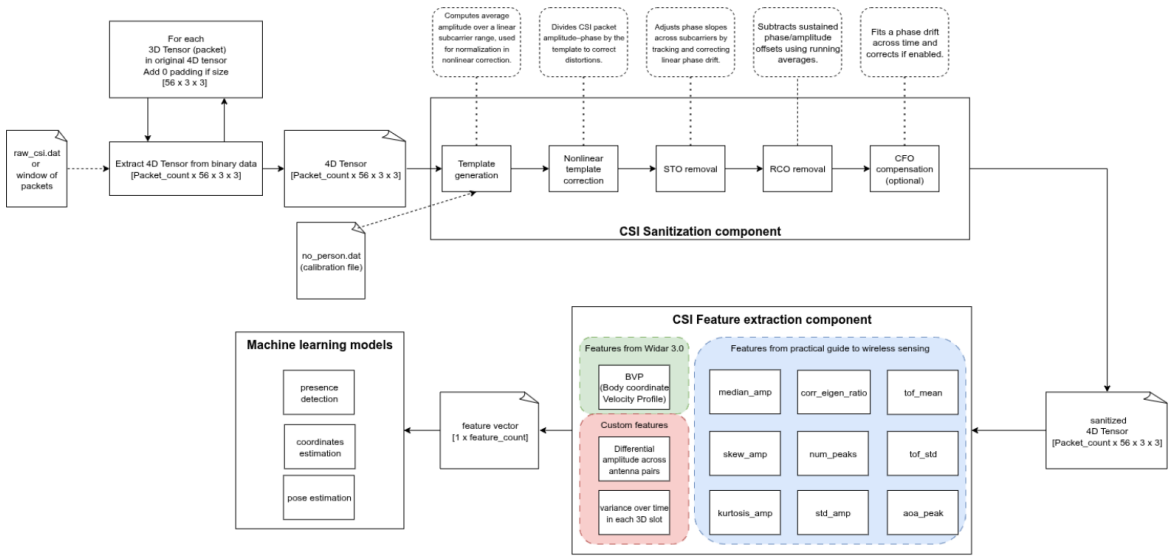

Figure 3: The main development pipeline, showing the stages of data sanitization and feature extraction before the machine learning models.

- **Nonlinear Template Correction:** Compensates for nonlinear distortions in the CSI data using empty room CSI data as a template.
- **STO/SFO Removal:** Mitigates phase shifts across subcarriers using a linear correction method.

- **RCO Removal:** Subtracts sustained phase/amplitude offsets using running averages.
- **CFO Compensation:** An optional step that corrects global phase rotation over time.

The output of this sanitization process is a clean, 4-dimensional tensor with dimensions (*packet_count*, *subcarrier_count*, *sending_antennas*, *receiving_antennas*), which in our setup was $(60, 56, 3, 3)$.

4.2 Feature Extraction and Models

After sanitization, we extract a vector of nine features designed to capture the physical and statistical characteristics of the Wi-Fi signal's interaction with a human body.

- **Physics-based features:** Time of Flight mean (tof_mean) and Angle of Arrival peak (aoa_peak).
- **Statistical features:** Amplitude Standard Deviation (std_amp) and Amplitude Skewness (skew_amp).

A Convolutional Neural Network (CNN) was used for all models because it works well for recognizing patterns and changes in the data over time. The models we developed include:

- **Human Presence Model:** This model uses an "Instability Footprint," a metric derived from the processed CSI, to distinguish an empty room from one with a human present.
- **Pose Classification Model:** This model classifies the human's pose based on patterns in the CSI data.
- **Localization Model:** This model estimates a person's approximate location within the room based on CSI data.

5 Findings and Challenges

The WiSentinel project successfully demonstrated that a system can detect human presence, classify poses and position using only Wi-Fi signals. Our custom-built pipeline proved to be a viable solution for this complex problem. However, the project also highlighted several significant challenges and areas for future improvement.

5.1 Difficulties Encountered

The most significant hurdle was the complete absence of existing tools and guidelines as well as the limitation of available data. Nearly every step of the process had to be custom made in order to function properly for our project.

- **Data Inaccuracy and Scarcity:** The limited time, manpower and available locations restricted the size and diversity of our dataset, which impacted the model's quality. More data, which can be easily recorded using our methodology, would fix this issue.

- **Environmental Interference:** As evidenced by our initial attempt in the library, Wi-Fi signals are highly susceptible to noise from other devices and materials. If this is not addressed and the data not properly sanitized, it will lead to severe data quality issues.
- **Hardware Constraints:** The project's success depended on a specific firmware and driver patch for specific hardware, which may not be transferable to every device. The problem stems from vendors who actively block access to raw wi-fi signal and custom firmware installation.
- **No pretrained ML models and methodology:** There are no publicly available pretrained models for classifying poses from CSI signals. There is no thorough description for model training methods either (what type of model to use, what data to use for inference and training, how much data to use and etc.). Feeding raw csi tensors also proven to be ineffective so feature extraction step was added. Not all 9 features from a feature vector can be used for each model. Depending on a model's purpose and related physical properties, specific features were used. Using redundant features can overtrain model or introduce noise which can confuse a model. E.g. features like time of flight, angle of arrival are useful for coordinates model.
- **Putting everything together:** Although Wisentinel is mostly PoC, it is still very complicated and contains many components from different domains: networking (wireless hardware configuration and application programming), machine learning (with multiple models describing different physical events: human presence, pose, coordinates), signal processing. To build this kind of system it requires careful planning, architecture design and a lot of testing.

6 Conclusion

The WiSentinel project stands as a successful proof-of-concept for a privacy-preserving human sensing system based on Wi-Fi CSI. We successfully custom-built an end-to-end solution that addresses a significant gap in non-intrusive monitoring. Our system is capable of detecting human presence, classifying poses, location in a way that is robust to environmental factors like no light, making it a powerful alternative to camera-based solutions.

The project's greatest contribution are the new tools and methodology we developed, from the custom firmware and data collection protocols to the signal processing pipeline and CNN models. By building our own tools, we have laid a foundation for future work in the field and made the work of potential future teams easier.

The initial idea of a project was to utilize already existing networking infrastructure. Businesses, government, common people won't have to by and install any extra hardware. Routers are ubiquitous in modern world and don't make people nervous like cameras. We need to detect and record people for medical purposes (someone fell and needs help), intrusion detection, movement in physical shops (very valuable data for companies like Lidl).

Wisentinel can solve all of this without requiring special hardware. All of data processing and machine learning can happen on any computing infrastructure, the only challenge is figuring out how to get access to csi data on different router models. OpenWRT is really

good opensource OS for routers which can help with that. It's compatible with many popular router models. Although even Openwrt requires tweaking to make csi data available but with clear generalized methodology it is absolutely solvable.

Future improvements could focus on

1. **Improving data collection methodology**, by introducing kinect and pose estimation model from camera to automatically label the data (which pose, orientation, position).

2. **Expanding the dataset**, by adding more poses and making it work with multiple people in the room.

3. **Defining more features**, by focusing on physical properties of a signal depending on a task, which model needs to solve.

4. **Generalizing solution for different router SoC**, by building custom csi-enabled Openwrt images and developing separate modules for csi data extraction.

Bibliography

References

[1] Zhao, J., et al. "WiDar 3.0: A WiFi-based Multi-Person 3D Pose Estimation System." https://tns.thss.tsinghua.edu.cn/widar3.0/data/TPAMI_Widar3.0_paper.pdf.

[2] Yan, H., et al. "Person-in-WiFi 3D: End-to-End Multi-Person 3D Pose Estimation with Wi-Fi." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. https://www.researchgate.net/publication/343456392_Compressed_Volumetric_Heatmaps_for_Multi-Person_3D_Pose_Estimation.

[3] OpenWrt. "How to install openwrt on TPLink archer a7." https://openwrt.org/toh/tp-link/archer_a7_v5.

[4] Seemoo-lab. "Atheros CSI Tool." https://github.com/seemoo-lab/nexmon_csi.

[5] Citysu. "csiread." https://github.com/citysu/csiread.

[6] Chen, Z., et al. "A CSI Dataset for Wireless Human Sensing on 80 MHz Wi-Fi Channels." *arXiv preprint*, 2023. https://arxiv.org/pdf/2305.03170.

[7] Guoxuan, G. "Hands on guide on wireless sensing." https://tns.thss.tsinghua.edu.cn/~guoxuan/assets/pdf/Paper-Hands-On.pdf.

[8] Nesbitt, I. "shake-UDP-live." https://github.com/iannesbitt/shake-UDP-live/blob/master/shake_udp_live.ipynb.

[9] Shuspieler. "Athero-CSI-tool-Python-RemoteReceive-Liveview-AmplitudeScaled." https://github.com/shuspieler/Athero-CSI-tool-Python-RemoteReceive-Liveview-AmplitudeScaled/tree/master/Client-Router.