SeismoAI: AI-Powered Seismometer

Yessenia Nicacio, Charlie Yonkura, Humza Ali, Kadidjatou Yattassaye, and Jafar Saniie

Embedded Computing and Signal Processing (ECASP) Research Laboratory (http://ecasp.ece.iit.edu)
Department of Electrical and Computer Engineering
Illinois Institute of Technology, Chicago, IL, U.S.A.

***Abstract*—This paper presents the design and implementation of an AI-powered seismometer designed to deliver faster, smarter, and more reliable earthquake detection. SeismoAI integrates seismic sensors, GPS localization, and environmental monitoring to generate real-time alerts that support disaster response and help save lives. The system has a companion mobile application that enables users to monitor device status, control system functions, and receive immediate earthquake notifications, transforming raw seismic data into information for earthquake analysis and hazard prediction. SeismoAI is a modular device and has an on-board AI pipeline powered by a TensorFlow Lite framework that converts the raw data into waveforms for real-time analysis. Our proposed system can be deployed in earthquake-prone regions as well as environments where abnormal vibrations pose a safety risk, such as mining operations or near construction sites.**

Keywords— AI-powered seismometer, IoT sensing systems, real-time alert systems, earthquake detection

I. Introduction

Seismic activity, whether originating from tectonic plate interactions or induced by human activities, such as mining, wastewater injection[1], or large-scale construction, poses a significant threat to infrastructure and human safety [2, 3]. Traditionally, monitoring these events has relied on high-precision broadband seismometers. However, the high cost and logistical complexity of these instruments often result in sparse monitoring networks, particularly in remote or developing regions [4]. In recent years, the emergence of MEMS (Micro-Electro-Mechanical Systems) accelerometers has transformed seismic monitoring, where these sensors offer a cost-effective and compact alternative, enabling the deployment of dense, large-scale wireless sensor networks. While these low-cost platforms have proven capable of recording local and regional seismic events, they are not without significant drawbacks. Documented limitations include higher self-noise levels, reduced sensitivity to low-magnitude events, and a heavy reliance on network-based synchronization (NTP), which can introduce timing uncertainties in critical scenarios [5].

A critical gap in current low-cost seismic solutions is their operational dependence on centralized processing. Most existing systems act as simple data loggers that transmit raw waveforms to off-site servers for analysis. This architecture introduces two major vulnerabilities:

- Latency: The time required for data transmission and centralized interpretation can exceed the narrow window available for Earthquake Early Warning (EEW).
- Connectivity: In seismic zones, major events often destroy network infrastructure first. This failure renders "cloud-dependent" sensors **inoperative** when they are needed most; therefore, reliable disaster monitoring systems must prioritize local processing and data analysis.

As established by the fundamental principles of EEW first proposed by Heaton in 1985 [6], the effectiveness of a warning system depends on the ability to detect the faster, weaker P-waves and issue an alert before the more destructive S-waves arrive. Every millisecond of processing saved at the "edge" (on-site) translates into critical seconds for automated responses, such as shutting off gas lines or slowing high-speed trains.

To address these challenges, this paper presents SeismoAI (shown in Figure 1), a portable, AI-enabled seismic system designed for autonomous edge operation. Unlike conventional low-cost sensors, SeismoAI integrates sensing, GPS localization, and environmental monitoring with an on-board deep learning pipeline. By leveraging the TensorFlow Lite framework, the system performs real-time waveform analysis directly on the device. This edge-computing approach allows for:

- Autonomous Detection: Immediate identification of seismic P-waves without requiring a cloud connection.
- Reduced Latency: Instantaneous alert generation at the source of detection.
- Rapid Deployment: A modular, "all-in-one" design optimized for quick installation in diverse and remote environments.

Figure 1. SeismoAI Device

979-8-3315-6026-3/26/$31.00 ©2026 IEEE

II. SeismoAI System Design

A. Hardware

SeismoAI hardware consists of a Raspberry Pi Model 4 B and a data acquisition front end based on an ESP32 microcontroller. We separate the functions of data acquisition and AI signal processing into two separate devices, primarily to ensure consistent sampling time when reading data from the ADCs. Figure 2 illustrates a high-level overview of the SeismoAI hardware and communication.

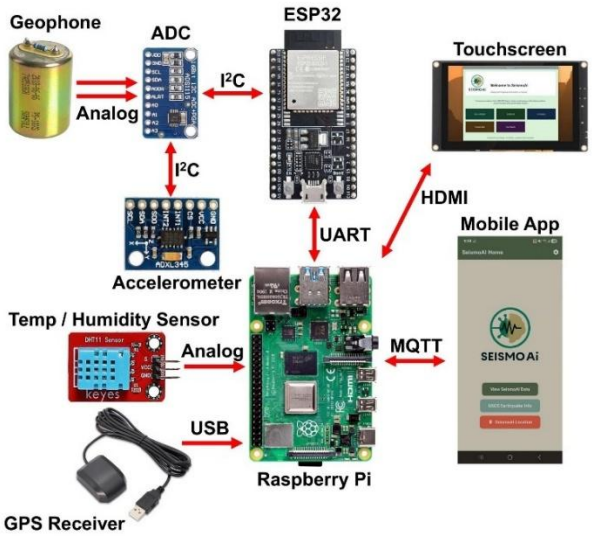


Figure 2. SeismoAI Diagram of Hardware and Communications

Technical components of the paper: The Raspberry Pi handles environmental data collection, AI signal processing, display functions, and Wi-Fi communication to our mobile app. The Raspberry Pi Model 4 B was selected because of its increased RAM compared to other models, which allows for more memory to be dedicated to running local AI models for seismic signal processing. The DHT11 temperature/humidity sensor is connected to the Pi for the collection of environmental condition data. Reading data from the DHT11 is not time-critical, so it can be done on a background thread on the Pi instead of the ESP32. A standard USB GPS receiver is connected to the Pi, which allows for precise timing data and latitude/longitude data to be collected. A standard USB Wi-Fi adapter is connected to the Pi to serve as the network over which MQTT messages are sent to the mobile app. Finally, a 5-inch touchscreen module was connected to control and view data from the Pi. The touchscreen module has threaded standoffs designed to match holes on the Raspberry Pi PCB, allowing easy connection of the Pi to the back of the screen.

The data acquisition front end, controlled by the ESP32, collects seismic data, formats the data into messages, and sends the messages to the Raspberry Pi. The seismic sensor used is the LGT-10HV-G20 bidirectional geophone. This geophone follows the industry standard design of a spring-mounted coil moving around a magnet in response to seismic activity. Two wires from the geophone connect to two differential ADC channels of the ADS1115 ADC module. The ADS1115 module was selected for its 16x programmable-gain amplifier (PGA) and its wide availability. The ADS1115 was configured to collect seismic data from the geophone at its maximum sampling rate of 860 SPS. This design choice was made to preserve waveform fidelity and maximize observable signal bandwidth of collected data. Although much of the relevant seismic energy is concentrated below 200 Hz, the higher sampling rate avoided premature bandwidth limitations during prototype development. Future iterations of SeismoAI may employ lower sampling rates, such as 400 SPS, to reduce computational load, storage requirements, and power consumption while achieving a sufficient Nyquist frequency to prevent aliasing of seismic wave detection. The ESP32 also collects data from an ADXL345 MEMS accelerometer to provide a comparison of data against the geophone data. The ADXL345 was chosen because it is a commonly available MEMS accelerometer. The ADS1115 and ADXL345 are both connected as targets on an I^2C bus, while the ESP32 serves as a controller on that bus. Figures 3 and 4 show a breadboard prototype of the system, followed by the protoboard of the data acquisition front-end circuit.

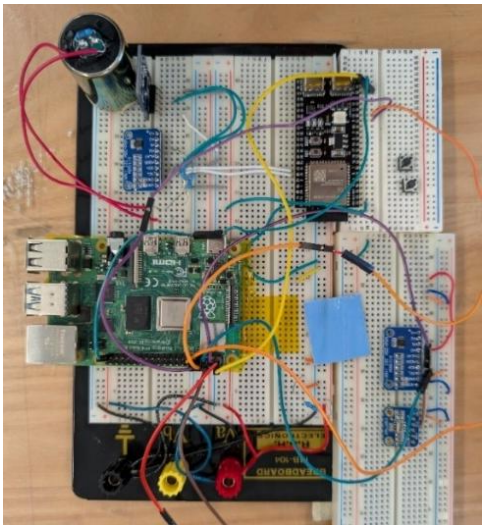


Figure 3. SeismoAI Breadboard Prototype

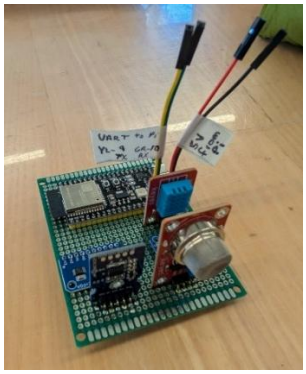


Figure 4. Data Acquisition Front End Protoboard

To ensure portability and structural integrity, the SeismoAI System is housed in a custom-designed enclosure, as illustrated in Figure 1. This housing integrates 3D-printed top and bottom plates with finger-jointed 1/4-inch acrylic

side panels. The internal architecture features dedicated standoffs for the protoboard and a specialized holder to secure the geophone, while the upper assembly aligns the touchscreen interface flush with the lid.

Furthermore, the enclosure incorporates functional cutouts for a power button and USB expansion. The transparency of the acrylic panels allows for real-time monitoring of the internal hardware and connection status. This integrated design facilitates the operational workflow of the system, providing a stable platform for data acquisition and visualization, as further detailed in Figure 5.

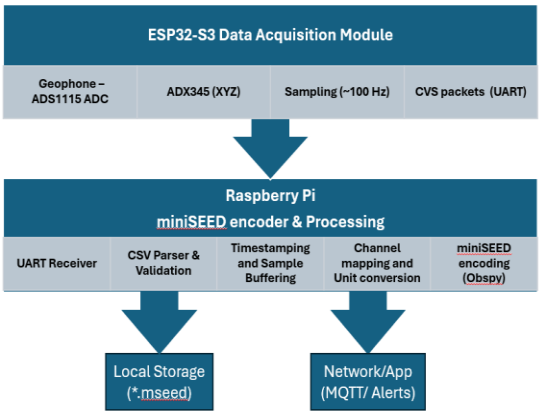


Figure 5. Data Conversion

B. Software

The SeismoAI software stack is designed to support real-time seismic data acquisition, local signal processing, AI inference, and user interaction. The architecture follows a modular pipeline, as illustrated in Figure 6, which outlines the data flow from raw input to cloud transmission.

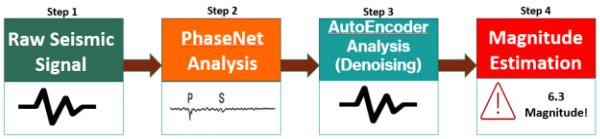


Figure 6. SeismoAI Pipeline flow

Ground sensor data is collected and preprocessed using the ObsPy framework, where filtering and formatting operations are applied before the data is passed to the Raspberry Pi's onboard AI module. Processed results are stored in mSEED format and subsequently analyzed by the AI engine. A comprehensive breakdown of the software components, including the frameworks used for machine learning and the user interface, is provided in Table I.

PhaseNet [7,8] is a deep neural network-based seismic phase picker that uses three-component waveform data to estimate probability distributions for primary wave (P-wave) arrivals, secondary wave (S-wave) arrivals, and noise. Arrival times are extracted from peaks of data received, enabling accurate and automated phase picking without manually entering data. PhaseNet demonstrates improved accuracy over traditional methods, particularly for S-wave detection [7].

The PhaseNet version used in this project is publicly available on GitHub [8].

In addition to using PhaseNet for phase picking, SeismoAI also utilizes an autoencoder-based model [9,10] for seismic anomaly detection. Autoencoders are unsupervised neural networks trained to learn a compact variety of normal input data and reconstruct the signal with minimal errors. When presented with anomalous data or previous signals, unseen signals, the reconstruction error increases, which then identifies abnormal seismic behavior. The autoencoder architecture used in this work is adapted from available PyTorch implementations. This secondary validation step helps mitigate residual noise and reduce anomalies from the data sets.

Table I. SeismoAI Software Stack Components

Category	Software	Purpose
Data Processing	ObsPy	Cleans, filters & processes data file formats before analysis
AI & Machine Learning	Phasenet & Autoencoder	Detect and classify seismic phases & detect anomalies
Communication	I2C, UART & MQTT	Enable real-time data transfer between sensors, Raspberry Pi & cloud
User Interface	Kivy/Flutter	Builds Raspberry Pi touchscreen interface for live sensors, GPS and AI Outputs
Data Sources	USGS	Provide real seismic datasets for testing & validation

- **Pipeline implementation:** Incoming seismic signals are segmented into fixed-length time windows and formatted before entering the AI model. As shown in Figure7, PhaseNet is executed first to generate P-Wave and S-Wave arrivals [11]. For those signals that exceed the predefined probability threshold, they are passed to the autoencoder for validation. The autoencoder reconstructs a denoised version of the signal. Only detections that satisfy both PhaseNet and Autoencoder are forwarded to create magnitude estimation, visualization, and logging stages.

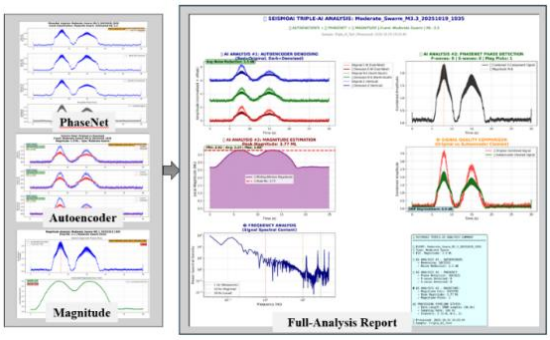


Figure 7. SeismoAI integrated Ai analysis pipeline output.

Furthermore, the SeismoAI pipeline was trained with seismic data obtained from the United States Geological Survey (USGS) [12], as well as additional datasets supplied by a geophysics researcher at the University of Illinois Urbana-Champaign. These datasets include three-component seismic recordings of real Earthquakes, allowing the design of a pipeline that has been tested under realistic conditions rather than solely on synthetic data.

While the current AI pipeline produces meaningful seismic detections and visual outputs, further refinement is required to improve robustness and generalization across diverse deployment conditions. Given the project's timeline, the combinations of both PhaseNet and Autoencoder demonstrate the feasibility of integrating both models for real-time analysis.

C. Mobile Application and User Interface

The SeismoAI mobile application was developed using the Flutter Framework and implemented in the Dart programming language, which allows cross-platform compatibility. The system architecture for this application is illustrated in Figure 8. The mobile application is designed to provide real-time alerts when the deployed device detects an earthquake or abnormal ground motion. When the detection of an event occurs, the seismometers onboard the Raspberry Pi 4B publish the seismic event data, and the mobile application that subscribes to this data stream receives live notifications that an event is occurring.

Figure 8. Mobile Application Architecture

In addition to the mobile application, SeismoAI features an onboard dashboard developed using the Kivy framework, as shown in the interface layout in Figure 9. This user interface allows users to view seismic data directly on the device without needing to return to a centralized workstation. Users can also access previous events on-site and initiate or stop data collection as needed. This enables immediate situational awareness during field deployments.

III. Experimental Results

To validate the functionality of the SeismoAI System, a field test was conducted using a portable battery pack to power the system (See Figures 10 and 11). Furthermore, a 3D-printed holder was used to secure the geophone in place. Using this test setup, all the following features have been tested:

- Real-time seismic data acquisition
- Proper storage of seismic recordings in MSEED format
- Remote operation via mobile application
- Electrical Safety

Figure 9. SeismoAI dashboard screen

The validation phase shows the following:

- The SeismoAI device successfully collected seismic data and stored them in MSEED format.
- The collected data is compatible with the downstream AI analysis workflow.
- The AI models (PhaseNet and Autoencoder) were loaded locally and operated without error during post-processing.
- The touchscreen interface was able to locally trigger scans and display recorded event magnitudes based on processed seismic data.

Figure 10. SeismoAI prototype setup

Figure 11. Touchscreen display showing recorded output. Note: Components were tested outside the final enclosure to confirm functionality before assembly

Although direct power consumption data was not measured during the test, the estimated system power consumption was derived from manufacturer specifications. The majority of the power consumption was due to the Raspberry Pi and the touchscreen module, while the sensors and data acquisition front end consumed comparatively little power. The total system power consumption is estimated to range between approximately 5-12 W depending on AI computation status, communication activity, and display utilization. The system was successfully operated from a standard 10,000 mAh USB battery bank during field testing, which demonstrates the feasibility of portable deployment.

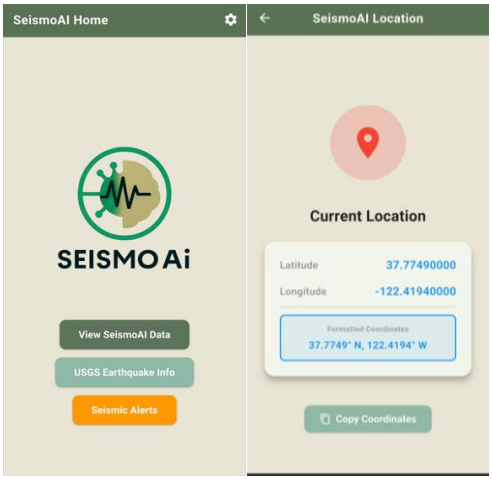
A separate analysis of the seismic data analysis AI pipeline was performed to measure the inference latency of the AI pipeline. First, to obtain preliminary inference latency metrics, two files were analyzed using our AI pipeline, and the time from the beginning to the end of AI model prediction was measured. Figure 12 shows the latency results for the system.

Data Source	# of Samples	Data Collection Period	Latency
SeismoAI Data Collection	1,154	30 sec	17 sec
USGS Published Data	270,000	2,700 sec	322 sec

Figure 12. Preliminary Latency Analysis Results

Next, a preliminary comparison of the magnitude estimation of our system against published USGS magnitude values was done. First, data of an earthquake in Silver Springs, Nevada was obtained from the NV31 seismic station, then processed through the AI pipeline without runtime errors. Significant deviation between the result of magnitude estimation and the published local magnitude value was observed, indicating that additional calibration and further development of the AI models is required before reliable magnitude estimation can be achieved.

A functional Android application was developed to enhance system control and field usability. The following features in Figures 13(a,b,c,d) illustrate key features found in the SeismoAI mobile interface:



Figures 13(a,b,c,d) (a) Home Screen provides access to SeismoAI data, USGS Earthquake Info, and Seismic Alerts, (b) The current coordinates of the SeismoAI device are displayed, supporting precise geotagging of seismic data, (c) The USGS Earthquake Map displays recent global seismic events along with the magnitude and timestamp, and (d) The settings screen includes options for Wi-Fi and Bluetooth configuration and provides remote control functions to power off and restart the digital seismometer.

IV. Conclusions

In this paper, a functional prototype of a smart, AI-powered seismometer was developed and tested using field-collected seismic data. We successfully demonstrated seismic data acquisition from a geophone using an ESP32, followed by processing through a three-step AI pipeline consisting of PhaseNet, an autoencoder, and magnitude estimation models running locally on a Raspberry Pi. A complete product design was carried out, including system architecture design, app development, AI model development, circuit design, protoboard assembly for data acquisition, enclosure design, product assembly, and functional validation.

SeismoAI was developed as a prototype and consequently, all quantitative performance evaluations are presented as a result of preliminary testing and are for exploratory purposes. However, our inference latency of 17 seconds for 30 seconds of collected data demonstrates the feasibility of

performing local seismic AI analysis on embedded hardware without relying on centralized cloud processing. Also, our estimated power consumption of 5-12 W supports the goal of rapid deployment in remote areas.

Our data collection and processing methodology produced results compatible with existing industry-standard analysis techniques in geology. A preliminary test of the AI pipeline resulted in magnitude estimations, however there were significant deviations from the published magnitude values, indicating that further development of the AI pipeline is necessary. However, our tests resulted in no runtime errors which demonstrate the potential to achieve the goal of autonomous local detection given further pipeline development.

SeismoAI is a scalable and reconfigurable system that can accommodate both minor modifications and major structural enhancements. The AI models employed can be further refined and expanded. To support more computationally demanding AI models, upgrading from a Raspberry Pi 4 Model B to a Raspberry Pi 5 may be necessary. SeismoAI also has the potential to be deployed as a distributed mesh-network alert system using long-distance radio communication technologies such as LoRa. This mesh network could be deployed in regions lacking earthquake monitoring and alert infrastructure, providing a low-cost solution in areas where the impact could be significant.

Field deployment of SeismoAI devices would require the integration of a solar panel and battery system to enable autonomous operation. The enclosure would need to be redesigned to ensure full weatherproofing. Additionally, SeismoAI could be scaled into a manufacturable product by designing a custom PCB to integrate all electronics, excluding the touchscreen and Raspberry Pi. The enclosure would also need to be optimized for cost, size, and environmental durability. Finally, collaboration with geophysicists to support research and deployment should be expanded, along with partnerships with government agencies capable of implementing this system at scale.

Acknowledgements

The authors would like to acknowledge the Grainger Foundation for granting SeismoAI the 2nd Place Award in the Grainger Computing Innovation Prize in support of this project.

References

[1] A J. I. Walter, P. Ogwari, A. Thiel, F. Ferrer, I. Woelfel, J. C. Chang, A. P. Darold, and A. A. Holland, "The Oklahoma Geological Survey statewide seismic network," *Seismological Research Letters*, vol. 91, no. 1, pp. 611–621, 2019, doi: 10.1785/0220190211.

[2] M. S. Abdalzaher, M. Krichen, and F. Falcone, "Emerging technologies and supporting tools for earthquake disaster management: A perspective, challenges, and future directions," *Progress in Disaster Science*, vol. 23, p. 100347, 2024, doi: 10.1016/j.pdisas.2024.100347

[3] G. Mei, N. Xu, J. Qin, B. Wang, and P. Qi, "A survey of Internet of Things (IoT) for geohazard prevention: Applications, technologies, and challenges," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4371–4386, May 2020, doi: 10.1109/JIOT.2019.2952593.

[4] M. Esposito, S. Marzorati, A. Belli, C. Ladina, L. Palma, C. Calamita, D. Pantaleo, and P. Pierleoni, "Low-cost MEMS accelerometers for earthquake early warning systems: A dataset collected during seismic events in central Italy," *Data in Brief*, vol. 53, p. 110174, Apr. 2024, doi: 10.1016/j.dib.2024.110174

[5] R. E. Anthony, A. T. Ringler, D. C. Wilson, and E. Wolin, "Do low-cost seismographs perform well enough for your network? An overview of laboratory tests and field observations of the OSOP Raspberry Shake 4D," Seismological Research Letters, vol. 90, no. 3, pp. 1380–1388, 2018, doi: 10.1785/0220180251.

[6] Heaton T. H. (1985). A model for a seismic computerized alert network. *Science (New York, N.Y.)*, *228*(4702), 987–990. https://doi.org/10.1126/science.228.4702.987

[7] W. Zhu and G. C. Beroza, "PhaseNet: A deep-neural-network-based seismic arrival time picking method," *Geophysical Journal International*, vol. 216, no. 1, pp. 261–273, Jan. 2019, doi: 10.1093/gji/ggy423.

[8] AI4EPS, "PhaseNet: A deep neural network for seismic phase picking," GitHub repository. [Online]. Available: https://github.com/AI4EPS/PhaseNet

[9] Lightning AI, "Deep Autoencoders," PyTorch Lightning Documentation. [Online]. Available: https://lightning.ai/docs/pytorch/stable/notebooks/course_UvA-DL/08-deep-autoencoders.html

[10] A.A Neloy and M. Turgen, "A comprehensive study of auto-encoders for anomaly detection: Efficiency and trade-offs," *Machine Learning with Applications*, vol. 17, Art. no. 100572, 2024, doi: 10.1016/j.mlwa.2024.100572.

[11] R. Sleeman and I. Van Eck, "Robust automatic P-phase picking: An online implementation in the analysis of broadband seismogram recordings," *Physics of the Earth and Planetary Interiors*, vol. 113, nos. 1-4, pp.265-275, 1999.

[12] United States Geological Survey, "USGS Earthquake Catalog and Seismic Data," [Online]. Available: https://earthquake.usgs.gov/earthquakes/search/