A Hybrid Swarm Intelligence Framework for Analysing Seismic Activity with Machine Learning

Sania[1]
School of Computer Science and Engineering
VIT-AP University
Andhra Pradesh, India
saniasilgawa81015@gmail.com

Riddhima Tripathi[2]
School of Computer Science and Engineering
VIT-AP University
Andhra Pradesh, India
riddhimatripathi02@gmail.com

Abstract- **Predicting earthquakes and assessing their associated risks is a complex challenge due to the unpredictable nature of seismic activity, the presence of noisy data, and the difficulty in identifying meaningful patterns across large datasets. Other statistical models like Poisson forecasting or the Gutenberg-Richter law often oversimplify seismic behavior while conventional machine learning models lack interpretability and domain-specific insight. In this paper, we address these issues by developing a hybrid machine learning framework that combines swarm intelligence techniques with domain-specific feature engineering. Instead of relying on raw data alone, we used Particle Swarm Optimization (PSO) to identify the most impactful seismic features and Ant Colony Optimization (ACO) to uncover the relationships between them, showing how seismic energy might travel through the Earth. Denoising is done using Kalman Filters, and Isolation Forest is used to spot rare earthquake events. With these inputs, a PSO-tuned XGBoost model was trained to predict earthquake magnitudes, while a Random Forest classifier assigned each event to a risk zone: Low, Medium, or High. The novelty of this approach lies in how all these pieces come together and that's what this paper discusses.**

Keywords: **Swarm Intelligence Techniques, Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Kalman Filter, XGBoost.**

I. INTRODUCTION

Picture living in Japan, where the ground can just decide to throw a tantrum any day - sometimes it's a little shake, sometimes it's a full-on disaster like the 2011 Tohoku quake that took over 15,000 lives and caused a jaw-dropping $360 billion in damage [1]. Japan's stuck on the Pacific Ring of Fire, where four tectonic plates are constantly brawling, spitting out about 1,500 quakes a year. That's 20% of the world's big ones (magnitude 6.0+), hitting packed cities like Tokyo and threatening everything from homes to nuclear plants. As of May 2025, Japan's still on edge, and figuring out when the next quake's gonna hit, how bad it'll be, and where to watch out is a massive challenge. It's not just science,it's about keeping people safe.

These days, with tons of seismic data pouring in from sensors and satellites, trying to spot patterns manually just isn't practical. That's where our framework steps in using swarm intelligence not just to boost performance, but to stay flexible and explainable. By blending optimization, interpretability, and machine learning into one smooth pipeline, we're tackling the shortcomings of older approaches. Seismology might be grounded in complex physics, but the real challenge today is turning that science into something useful tools that don't just help researchers, but also guide decision-makers, emergency teams, and everyday people when it matters most.

Traditional methods, like the Gutenberg-Richter law, use basic math to study earthquakes but miss their complex patterns. Many machine learning tools predict quakes but don't explain their results clearly, making them hard to use for planning. A 2019 study by Abraham and Rohini used PSO to predict quake magnitudes in Japan, achieving good results but missing energy mapping or risk zoning [2]. A 2023 study used XGBoost to predict strong quakes in Japan, California, and Israel, helping warn about a 2022 Fukushima quake, but lacked swarm methods like PSO or ACO.

The proposed model is trained and validated on 2001–2018 Japan earthquake records from the USGS open-access seismic database, one of the most reliable and globally recognized sources for earthquake monitoring.The new approach derived by us uses PSO to pick key details like magnitude and depth, ACO to map quake energy flow, Kalman Filters to for noise reduction , and DBSCAN to spot unusual events, feeding an XGBoost model to predict quake sizes accurately and a Random Forest model to label areas as Low, Medium, or High risk perfectly .This framework's high accuracy and holistic approach make it a stronger tool for Japan's urgent seismic needs.

The rest of the paper is organized as follows. Section II presents the preliminaries, while Section III outlines the research methodology. Section IV illustrates and discusses our findings. Section V explains our approach to predicting the number of future earthquakes based on past trends. Finally, Section VI concludes the paper.

II. PRELIMINARIES

Swarm Intelligence. Swarm Intelligence refers to a class of nature-inspired, population-based algorithms within the broader domain of evolutionary computation [3]. These algorithms operate through a population of individuals, each representing a potential solution. Using decentralized cooperation and information sharing, the population collectively evolves over successive iterations, gradually improving solution quality and converging toward optimal or near-optimal outcomes. Research in swarm intelligence is typically categorized into three primary areas. The first key principle is decentralized control, as we discussed above. For example, in ant colony optimization (ACO), artificial ants deposit pheromones on paths they travel. Other ants sense these pheromones and probabilistically choose paths with higher concentrations, leading to the emergence of optimal paths without a central planner. Similarly, in particle swarm optimization (PSO), individual particles adjust their trajectories based on their own experience and the best-known positions of neighbouring particles. The second principle focuses on autonomy, in which behaviour is analyzed between agents. Like in flocking algorithms, there are three rules – Separation, that is maintaining distance between boids to avoid bumping into each other, Alignment referring that boids match their velocity (dx/dt) with their individual neighbours and Cohesion where boids try to stay closer to their neighbours tending towards the center of the group. The third principle emphasizes robustness and fault tolerance. Swarm systems are resilient because their durability doesn't depend on an individual agent, which decreases the chances of full system failure. For example, integration of the Firefly Algorithm in Fault-Tolerant Sensor Networks. In this system, the nodes are deployed in huge numbers, increasing the probability of failure due to battery loss or any physical damage. However, using the firefly algorithm nodes can self-organize by adjusting their positions based on neighbour nodes' brightness (fitness). Even if several nodes fail, the remaining nodes carry the responsibility, thus forming new communication paths and maintaining overall network functionality. This resilience emerges from their distributed nature and ability to adapt locally based on attraction rules. Though there are various swarm algorithms available but we decided to go with PSO and ACO for this study due to their effectiveness in feature optimization and pattern discovery within seismic datasets.

Particle Swarm Optimization. The optimization can be defined as a mechanism through which the maximum or minimum value of a given function or process can be found. The function that we try to minimize or maximize is called as objective function. Two key strategies drive the optimization process: exploration and exploitation. Exploration refers to the process of increasing the domain of knowledge regarding the environment or model. It involves choosing actions with uncertain outcomes to learn how they affect the environment and what rewards they might lead to. Exploitation means using what you already know to make decisions that are expected to give the best results. [4]

PSO is a technique that applies these principles, using a population of solutions that learn and adapt over time to solve complex problems efficiently. Imagine a flock of birds soaring through the sky, each one adjusting its path based on where it's been and what the flock's leader is doing.[10] That's where the core logic of PSO lies, a smart global optimization technique inspired by nature's swarms like birds, fish, or even ants working together. Developed in 1995 by James Kennedy and Russell Eberhart, PSO mimics this collective intelligence to tackle complex problems. In our study, we are using it to pick the best seismic features for predicting earthquake magnitudes. Picture each "particle" as a potential solution floating around in a multi-dimensional space, think of it like a map where each spot represents a combination of seismic features. The particles update their positions based on both their personal experience and the experience of the swarm, effectively balancing individual learning and social cooperation. At each iteration, every particle evaluates a fitness function at its current position, which represents how good the solution is. It then adjusts its trajectory based on three key components: inertia (maintaining current direction), personal best (its own best-known position), and global best (the best position found by the swarm)[5]. Random factors influence these updates to encourage exploration and avoid premature convergence. This is mathematically described as:

$$v_{t+1} = v_t + c_1r_1\,(g - x_t) – c_2r_2\,(p - x_t) \quad (1)$$

$$x_{t+1} = x_t + v_{t+1} \quad (2)$$

Here, x_t :the current position and v_t denote the current position and velocity of a particle at time t, p is the best position the particle has found so far, and g represents the best position identified by the entire swarm. The constants c_1 and c_2 are learning coefficients, while r_1 and r_2 are random values sampled uniformly from the interval [0,1], introducing stochastic behavior into the movement.

To enhance the performance of the basic PSO, Eberhart and Shi later proposed the inclusion of an inertia weight, leading to the modified velocity update:

$$v_{t+1} = \omega v_t + c_1r_1\,(g - x_t) - c_2r_2\,(p - x_t)$$

The constriction coefficient χ is calculated as:

$$\chi = \frac{2\kappa}{|2-\varphi-\sqrt{\varphi^2-4\varphi}\,|}$$

Here, κ is typically set to 1, and φ=c1+c2 , with $\varphi\leq4$ ensuring the square root yields real values. When φ=4, the algorithm reverts to the classical PSO model with χ=1.

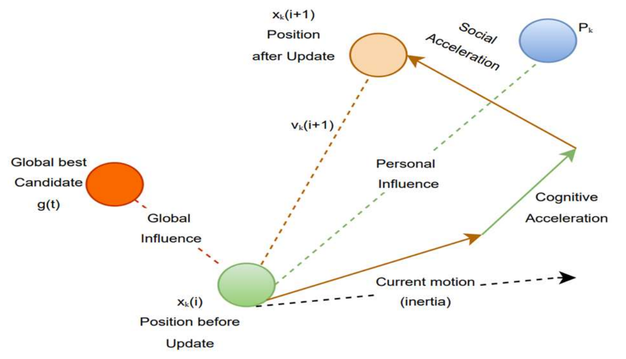

Figure 1. Velocity and Position Update Dynamics in Particle Swarm Optimization

Ant Colony Optimization. ACO is a population-based metaheuristic inspired by the natural foraging behavior of real ant colonies. It is a smart search technique uniquely inspired by how real ants find their way to food. Marco Dorigo first introduced it in 1992 to solve tough problems like the Traveling Salesman Problem. So, how do ants figure out the best route? In nature, ants leave behind a scent trail called pheromones as they walk. Pheromones encode a long-term memory about the whole ant search process. Other ants try to follow stronger trails, and if a particular path leads to food, more ants take it and reinforce it. Eventually, the shortest route becomes the most used and most reinforced. That's how the colony, without any leader, finds the most efficient path [6]. This emergent intelligence stems from decentralized coordination, where ants interact indirectly through a mechanism called **stigmergy,** which is a form of communication achieved by modifying the environment. This dynamic balance between reinforcement and decay helps the algorithm explore the solution space while avoiding premature convergence.

In this study, we used ACO to find out the best order or path among seismic features, after selecting the most important ones using PSO. Think of each seismic feature (like amplitude, energy, velocity) as a point on a map. It helps our artificial ants "walk" across these features, finding a path that makes the most sense, where features are strongly correlated but not redundant. After several iterations, the most efficient path emerges, which we then pass into the final prediction model. This helps capture not just which features matter, but how they interact, leading to better, smarter earthquake predictions. This application of ACO not only captures physical relationships between seismic variables but also enhances the performance of our downstream machine learning models. In ACO, each connection between nodes (denoted as l_{ij}) is associated with a pheromone level (represented as τ_{ij}). Also, each arc may carry a heuristic value η_{ij}, which provides prior knowledge or dynamically generated insights independent of the ants' behavior; this could be based on the problem's definition or real-time system feedback.

Some of the characteristics of ant colonies are:

- Although individual ants can construct their solution (probably poor) yet collective effort of all ants can provide optimized solutions.
- Each ant relies on its local knowledge and as there is no access to global information.
- Communication among ants is indirect. Instead of explicit messaging, they influence each other through the pheromone trails.[7]

Every ant k is equipped with a memory M^k that helps it keep track of its journey so far. This memory serves three primary functions: it assists in building solutions, in evaluating the quality of those solutions, and in retracing the path to integrate it with pheromones later. At any given moment, an ant in state $sr=\langle sr,i\rangle$ can transition to a new node j within its feasible neighborhood N_i^k, which includes all nodes that are both reachable from its current state and valid under the problem's constraints. Each ant starts from an initial state s^k and continues moving through neighboring states until a predefined termination condition ϵ^k is met. The decision of which node to move to next is made probabilistically.

As ants travel from node i to node j, they may update the pheromone trail on that connection τ_{ij} immediately which is called a behaviour known as online step-by-step pheromone update. This allows other ants to immediately benefit from potentially promising paths. Once a complete path has been constructed, the ants perform online delayed pheromone update in which they backtrack along the same route to reinforce the pheromones. If the path is of poor quality, the ant may explore alternative routes, freeing up the trail for better paths to emerge. ACO algorithms incorporate two additional components that enhance optimization performance: pheromone evaporation and daemon actions. Pheromone evaporation slowly reduces the intensity of pheromone trails over time, preventing the algorithm from prematurely converging to suboptimal solutions. It promotes exploration by allowing new paths to emerge, acting as a memory decay or forgetting strategy. On the other hand, daemon actions represent centralized control elements that single ants cannot perform. These actions include triggering local optimization routines or depositing extra pheromones selectively. This process known as offline pheromone updating. The ACO metaheuristic often employs a construct such as schedule_activities, which states three core operations: (i) ant solution construction, (ii) pheromone evaporation, and (iii) daemon actions. The schedule_activities framework does not rigidly define execution order or synchronization, giving the flexibility to adapt it for parallel implementations as needed.[8]

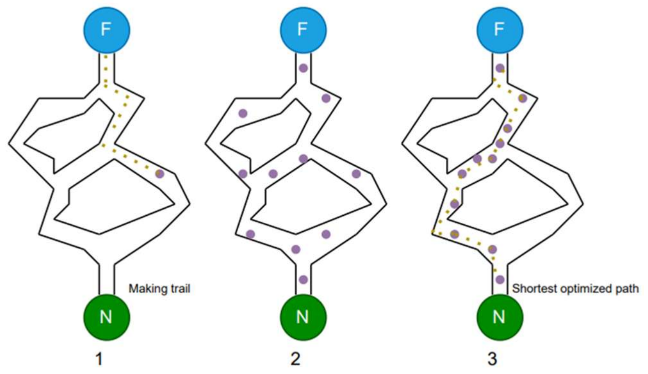

Figure 2. Illustration of Ant Colony Optimization

The three subfigures illustrate how ACO mimics ant behavior to discover optimal paths. In (1), ants begin exploring different routes between the nest (N) and the food source (F), leaving pheromone trails. In (2), more ants follow stronger pheromone paths, reinforcing them over time. Finally, in (3), the colony converges on the shortest and most efficient route, marked by the densest pheromone trail.

III. METHODOLOGY

In this section, we describe the methodology of the proposed study. We first introduce the dataset that forms the foundation of our analysis, followed by a detailed explanation of the preprocessing steps undertaken to clean the data**.**

Dataset. For this study, we used a dataset published by the United States Geological Survey (USGS). The USGS is a globally recognized scientific agency that continuously monitors seismic activity and provides reliable geophysical data. The dataset spans earthquake events across Japan from 2001 to 2018, including key seismic attributes such as latitude, longitude, magnitude, depth, timestamp, magnitude type, station ID, date, and time. Our approach begins with thorough data cleaning and denoising, followed by swarm intelligence-based feature optimization and machine learning driven prediction.

Data Ingestion. The first step involved collecting and importing seismic data into a structured format suitable for processing. We sourced real-time seismic waveform data in CSV format, which contained essential parameters such as timestamps, amplitude readings, station IDs, geographical coordinates (latitude and longitude), and earthquake magnitudes. To handle data, we utilized Python's Pandas and NumPy libraries. These tools helped us to efficiently read the CSV files and organize the data into dataframes for further cleaning and transformation steps. Proper ingestion guaranteed that we worked with the most recent and relevant seismic records, thereby increasing the reliability of our results.

Data Cleaning. Once the seismic data was loaded, the next step was to clean it, that is, to make sure it was accurate, consistent, and ready for analysis. Here we checked for missing values, duplicate entries, and any inconsistencies in the dataset. One of the key challenges encountered was dealing with missing columns in the dataset. In a few of the CSV files, crucial fields such as amplitude, depth, or even timestamp were either partially filled or completely absent. This created issues when trying to apply uniform preprocessing steps across the entire dataset. For example, without consistent timestamp data, it became difficult to calculate time-based features like acceleration or to synchronize readings from multiple stations. In some cases, column headers were mismatched or slightly renamed (e.g., "mag" instead of "magnitude"), which initially caused errors during parsing and aggregation. We manually inspected and standardized these headers to ensure smooth loading. Inconsistencies in value formats in some latitude and longitude values were in degrees, were handled using custom conversion functions written in Python. These functions standardized all geographic coordinates into decimal format, allowing consistent spatial analysis and feature extraction. Missing but essential columns led to dropping those specific records entirely, while partially missing rows were repaired using interpolation. These steps were critical to avoid noise or bias in the model due to incomplete or misaligned data. We calculated the mean of valid nearby coordinates (based on station ID or surrounding rows) for entries that were completely missing or unreadable.

Noise Reduction. After cleaning the data, some sudden spikes and drops were noticed, indicating some of the seismic amplitude values were noisy. To smooth out these irregularities without losing important signal information, we applied the Kalman Filter. This is a recursive algorithm that estimates the true value of a variable by filtering out random noise based on previous measurements. Traditionally, the Kalman Filter is derived by aiming to minimize the mean squared error, which serves as a measure of accuracy. However, it can also be interpreted through maximum likelihood estimation, offering a statistical perspective on its workings. The Kalman Filter operates in two main phases: Prediction and Update.

1. Prediction Step

State Estimate Prediction:

$$x_{k|k-1} = Ax_{k-1|k-1} + Bu_k$$

Error Covariance Prediction:

$$P_{k|k-1} = AP_{k-1|k-1}A^T + Q$$

2. Update Step

Kalman Gain Calculation:

$$K_k = \frac{P_{k|k-1}H^T}{HP_{k|k-1}H^T + R}$$

State Update:

$$x_{k|k} = x_{k|k-1} + K_k(z_k - Hx_{k|k-1})$$

Covariance Update:

$$P_{k|k} = (I - K_kH)P_{k|k-1}$$

where:

- x = state estimate,
- P = error covariance,
- A,B = state and control input models,
- Q,R =process and measurement noise covariance matrices,
- H = observation model,
- K_k = Kalman Gain,
- z_k = measurement at time kkk,
- I = identity matrix.

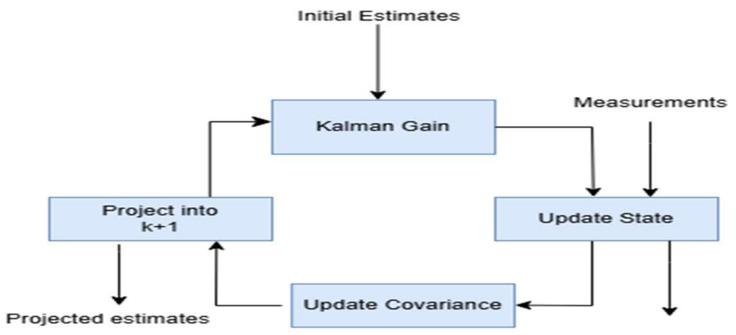

Figure 3. Kalman Filter Recursive Algorithm

Magnitude	Magnitude (Denoised)	Amplitude	Amplitude (Denoised)	Velocity	Velocity (Denoised)
5.20	5.20	1.58e+08	1.58e+08	1393.38	1393.38
4.80	4.80	6.30e+07	6.40e+07	879.16	884.14
4.86	4.78	7.24e+07	5.52e+07	942.04	847.16
4.94	4.84	8.70e+07	6.48e+07	1032.92	910.52
4.94	4.87	8.70e+07	7.04e+07	1032.92	945.66

Table 1. Comparison of Original and Denoised Earthquake Parameters

Such preprocessing is crucial when identifying high-magnitude events; in this case, **14,092 earthquakes** were found with a denoised magnitude above 9, indicating significant seismic activity after noise reduction.

Time-Series and Spatial Feature Analysis. We derived seismic frequency, acceleration, and velocity by applying derivative-based and signal processing techniques to the raw time-series signals by extracting dynamic features from the seismic data. These transformations allowed to capture the rate of change in ground motion, helping differentiate between low-impact and high-impact earthquakes[12].

Velocity:

$$v(t) = \frac{\mathrm{d}}{dt}[\text{Displacement(t)}]$$

Acceleration:

$$a(t) = \frac{\mathrm{d}^2}{dt^2}[\text{Displacement(t)}]$$

Frequency:

It is calculated using the Discrete Fourier Transform(DFT):

$$\mathrm{X_k} = \sum_{n=0}^{N-1} \mathrm{x_n} \cdot \mathrm{e}^{-\mathrm{j}\frac{2\pi}{N}\mathrm{kn}}$$

where:
X_k is the amplitude of the k-th frequency bin.
N is the total number of samples.
x_n represents the time-domain signal.
k denotes the frequency bin index.

After applying DFT, the actual frequency corresponding to each bin is computed as:

$$f_k = \frac{k \cdot f_s}{N}$$

where:

f_s : The sampling Frequency

f_k: Real world frequency of kth bin

To spatially categorize seismic activity, we converted the geographical coordinates of the seismic stations into discrete risk zones using geospatial binning. This was achieved by dividing the latitude and longitude values into 10 bins each and creating unique spatial labels for each bin combination. Each resulting zone represented a grid-based spatial classification unit. It enabled spatial clustering, which is used for risk mapping, emergency preparedness, and policy formulation in disaster management systems.

Feature Scaling. Normalization and standardization are very essential steps of feature scaling. Feature Scaling refers to the idea of preparing the data for ML models[9]. It ensures that all different features in the dataset are on a similar scale. For example, if one feature ranges from 1 to 10 and another from 1000 to 10000, the model might unintentionally give more weight to the larger values, even if they're not more important. This can result in biased predictions and slow down training. By applying feature scaling, it is ensured that all features contribute equally. Techniques like normalization and standardization are commonly used for this. Using Scikit-learn's StandardScaler feature scaling was applied in this paper through z-score standardization, making sure that key seismic attributes like magnitude, depth, and acceleration were transformed to a standard scale with mean 0 and unit variance. Well, standardization refers to first centering and then scaling of data.

The transformation formula applied is:

$$z = \frac{x - \mu}{\sigma}$$

where:

x = original value

μ= mean of the feature

σ= standard deviation of the feature

z= normalized value

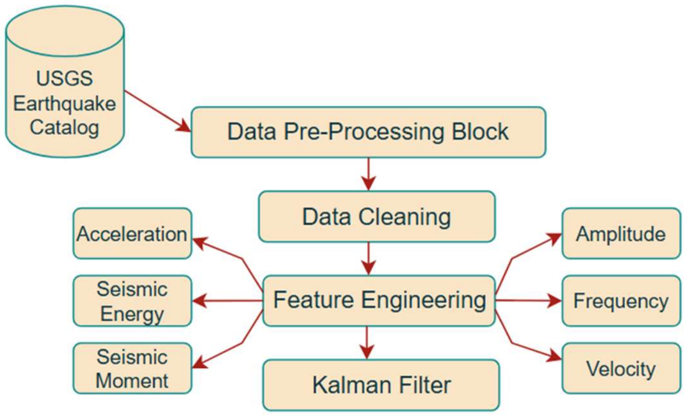

Fig. 4 Data Preprocessing Block

Anomaly Detection. An anomaly (also called an outlier) is a data point that is significantly different from the rest of the data**.** To strengthen the anomaly detection block of our earthquake analysis, we applied a hybrid strategy using both Isolation Forest and DBSCAN(Density-Based Spatial Clustering of Applications with Noise). Isolation Forest is an unsupervised machine learning algorithm that works by trying to isolate each data point. The logic is simple: if something's an outlier, it should be easier to isolate than the rest. The algorithm builds random trees, and data points that get isolated in fewer steps are likely anomalies. We fed in all the important numerical features like magnitude, depth, amplitude, velocity, and seismic energy, and let the model flag the events that didn't follow the usual pattern. With a contamination rate of 5%, the algorithm marked around 705 events as anomalies. These could be rare seismic events or even data glitches.

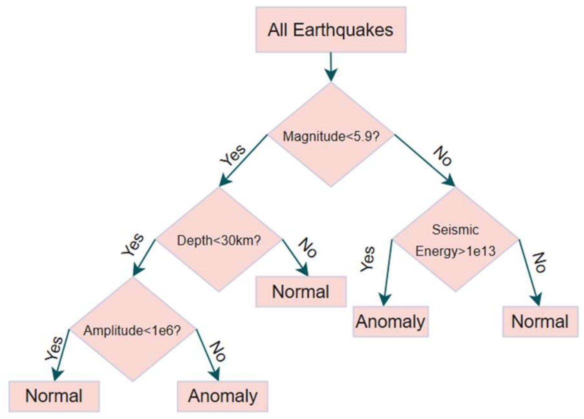

Fig 5. Anomaly Detection Flowchart

To illustrate the internal logic of Isolation Forest, we present a sample tree with assumed feature splits (e.g., Magnitude < 5.8, Depth < 35 km). These values are not extracted from the actual model but serve to demonstrate how the algorithm recursively isolates data points based on random feature-threshold combinations.

To complement the statistical anomaly detection of Isolation Forest, we applied DBSCAN to uncover spatial anomalie**s** in earthquake occurrences across Japan. By clustering data points based on their geographical coordinates (latitude and longitude), DBSCAN identifies regions of high seismic activity and flags isolated quakes as spatial outliers. These results helped us identify unexpected seismic zones, which could be critical in proactive disaster planning and early warning systems.

Feature Selection Using PSO. Since the core concepts and mechanics of PSO were explained earlier in the Preliminary section, this part focuses on how it was applied in our model and the results it achieved.

In the first phase of the pipeline, PSO was used to automatically identify the most informative seismic features from the dataset. This step helped reduce noise and eliminate less relevant attributes. PSO maximizes the mutual information between each feature and the target variable (magnitude), verifying that only those features that had strong predictive power were selected. The fitness function used for this was the average of mutual information scores across the selected features.

After running PSO, the algorithm consistently highlighted the following features as the most significant:

- ➢ **Amplitude**
- ➢ **Velocity**
- ➢ **Seismic Energy**
- ➢ **Seismic Moment**
- ➢ **Risk_Zone_encoded**

These features were chosen because they capture both the intensity of seismic events and their geographical impact, making them ideal inputs for accurate magnitude prediction.

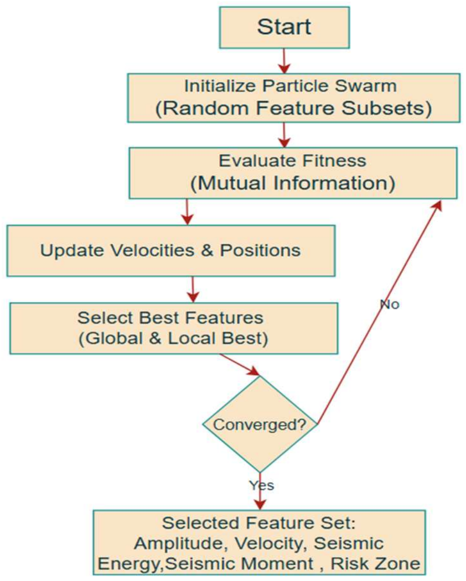
Start
Initialize Particle Swarm (Random Feature Subsets)
Evaluate Fitness (Mutual Information)
Update Velocities & Positions
Select Best Features (Global & Local Best)
Converged?
No
Yes
Selected Feature Set: Amplitude, Velocity, Seismic Energy,Seismic Moment , Risk Zone

Fig. 6 Working Of PSO

Feature ordering using Ant Colony Optimization. A detailed theoretical background and working principles of ACO have already been discussed in the Preliminary Section, providing the foundational context for its practical application in this study. Although PSO selected the best features, ACO was then used to determine their optimal ordering**,** representing the best path of influence among features, simulating how seismic waves propagate through them.

Each seismic station was selected as a node in a graph, and seismic wave propagation paths between these nodes were marked as edges. Artificial ants are initialized to traverse this graph, finding optimal paths based on pheromone intensity and Heuristic desirability that is calculated from seismic variables such as depth, frequency, and magnitude gradients. At each iteration, ants probabilistically selected their next node using exploration and exploitation (discussed in the preliminary section). After completing their paths, the ants deposited pheromones based on the quality (shortness and efficiency) of their paths, reinforcing effective seismic propagation routes.

XGBoost with PSO-Tuned Hyperparameters and ACO Selected Features. XGBoost (Extreme Gradient Boosting) is a highly efficient and scalable implementation of gradient boosting algorithms.[13]

In the proposed paper, XGBoost is used as the regression model to predict earthquake magnitudes based on swarm intelligence-selected seismic features. The model builds an ensemble of decision trees sequentially, with each new tree focusing on correcting the prediction errors made in previous model training. The steps are as follows:

1. **Initialization with a Base Learner:** The first decision tree is trained using the ACO- implemented seismic features selected through PSO. For regression tasks like earthquake magnitude prediction, this initial model typically predicts the average of the observed magnitudes across the dataset.

2.**Error Calculation:** After the initial prediction, the residuals (the differences between the actual earthquake magnitudes and the predicted ones) are calculated. These residuals represent the aspects of the data that the model has not yet captured.

3.**Sequential Tree Correction**: A new decision tree is trained specifically to predict these residuals. By learning from where the previous tree made errors, this tree helps re-evaluate the overall prediction. This process continues with each subsequent tree, aiming to minimize the residual errors further.

4.**Iterative Correction:** This cycle of error correction repeats for a defined number of boosting rounds (optimized using PSO), with each tree contributing a small adjustment to the model. The model becomes progressively better at capturing complex patterns in the seismic data.

5.**Final Prediction Aggregation:** The final output is obtained by aggregating the predictions of all the trees, essentially summing the predictions from each iteration. This ensemble approach ensures that the model has high predictive power and generalizes well across unseen earthquake events.

Mathematical Operations Behind XGBoost:

Objective Function:

$$\mathcal{L}(\phi) = \sum_{i=1}^{n} l(y_i, \widehat{y_i}) + \sum_{k=1}^{K} \Omega(f_k)$$

where:

y_i is the actual magnitude of the ithi^{th}ith earthquake,

$\widehat{y_i}$ is the predicted magnitude from the model,

$l(y_i, \widehat{y_i})$ is the loss function

Ω is the regularization term for the k-th tree

K is the total number of trees.

Loss Function:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2$$

Gradient and Hessian Calculation:

$$g_i = \frac{\partial l(y_i,\widehat{y_i})}{\partial \widehat{y_i}}, \quad h_i = \frac{\partial^2 l(y_i,\widehat{y_i})}{\partial \widehat{y_i^2}}$$

At each iteration, XGBoost computes the first-order (gradient) and second-order (Hessian) derivatives of the loss function.

Tree Structure and Split Finding:

$$\text{Gain} = \frac{1}{2}\left[\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda}\right] - \gamma$$

where:

G_L, H_L sum of gradients and Hessians on left child node

G_R, H_R : sum of gradients and Hessians on right child node

λ: L2 regularization term

γ: pruning parameter (minimum gain to split)

Final Prediction:

$$\widehat{y_i} = \sum_{k=1}^{K} f_k(x_i)$$

Random Forest Classifier for Earthquake Risk Zone Prediction.

In addition to predicting earthquake magnitude, this paper incorporates a Random Forest Classifier (RFC) to categorize seismic events into predefined **risk zones**: *Low*, *Medium*, or *High*, based on their magnitude and geophysical features.

Random Forest is an ensemble learning technique that builds a forest of decision trees, each trained on a random subset of the data and features. The final prediction is determined by a majority vote across all trees, making it highly robust to overfitting and noise.

All the results and insights of each and every algorithm is discussed in Results and Discussion section .

IV. RESULTS AND DISCUSSION

A. Geospatial Distribution of Earthquakes and Magnitude Analysis.

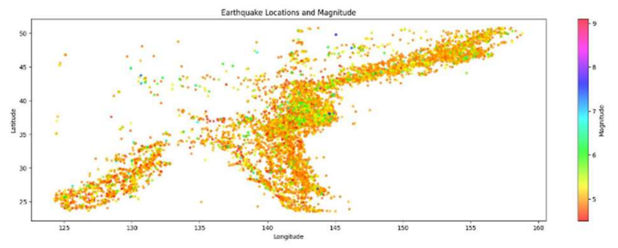

Fig 7. Distribution of earthquake events across geographic coordinates (latitude and longitude)

The scatter plot displays earthquake locations with magnitudes up to 9.1, color-coded using the HSV palette:

- Warm colors (red/yellow) indicate high-magnitude events.
- Cool colors (blue/purple) represent lower magnitudes.

Spatial clustering is observed, especially around tectonic plate boundaries and fault zones, highlighting regions of intense seismic activity

Depth and Magnitude Relationship in Earthquake Events

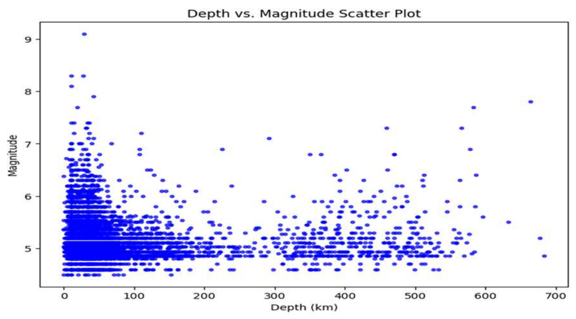

Fig 8. Depth vs Magnitude Scatter Plot

Fig. 8, shows each point on the plot corresponds to a unique earthquake event, with the x-axis representing the earthquake's depth and the y-axis representing its magnitude.

Here, shallow earthquakes (less than 70 km) are sometimes associated with higher magnitudes, deep-focus earthquakes, reaching depths near 700 km, can also exhibit significant magnitudes, particularly in subduction zones.

Magnitude Distribution of Earthquake Events

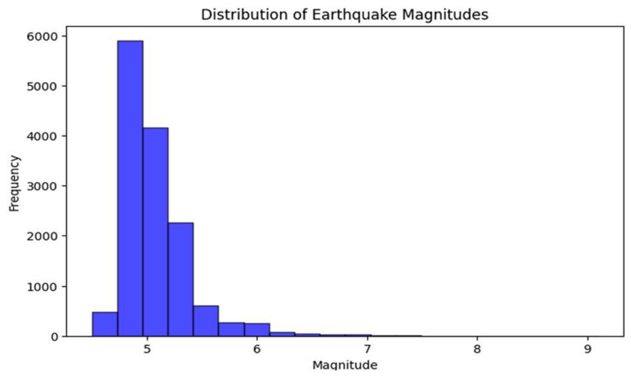

Fig 9. Frequency distribution of earthquake magnitudes

In Fig. 9, the histogram reveals that most earthquakes in the dataset fall within the magnitude range of 4.0 to 5.0, indicating that moderate seismic events are the most frequent. Notably, there is a single rare instance of a high-magnitude earthquake (9.1), which was the 2011 Tohoku earthquake that struck off the northeast coast of Japan on March 11, 2011.

B. Distribution of Earthquake Depths

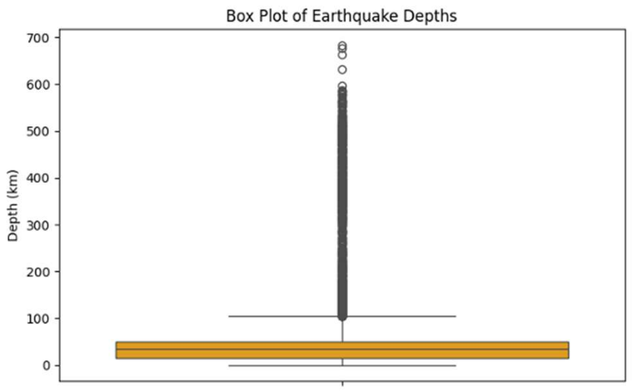

Fig 10. The Box Plot of Earthquake Depths

In Fig. 10, the y-axis represents the depth of the earthquake (in kilometers), while the box plot visualizes the minimum, first quartile (25th percentile), median, third quartile (75th percentile), and maximum depth.

The box plot of earthquake depths highlights that most seismic events occur within a moderate depth range, typically between 50 and 700 km, with the median and interquartile range (IQR) reflecting the central tendency and spread of the data. It also reveals several outliers, representing unusually shallow or deep earthquakes that may occur in unique geological settings like subduction zones or tectonic plate boundaries. While shallow to intermediate-depth earthquakes are more common and associated with active tectonic regions, deep-focus events nearing 700 km are rarer but significant.

C. Variation of Earthquake Magnitudes Over Time

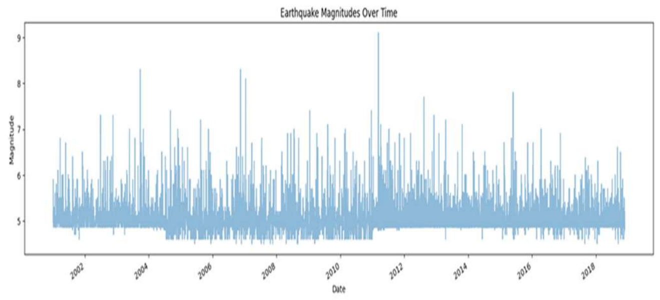

Fig 11. The Earthquake Magnitudes Over Time line Plot

Fig. 11 represents the x-axis represents the date of the earthquake events, while the y-axis shows the magnitude of each earthquake. The plot represents the data as a continuous line, with the magnitude values represented over time.

Each peak in the graph marks a period of increased seismic activity, while the flatter sections reflect quieter times with smaller tremors. By looking at these ups and downs, we can spot possible patterns, such as clusters of strong earthquakes or recurring cycles. This kind of visualization helps to monitor seismic trends and may support better forecasting of future events.

D. Relationship Between Earthquake Magnitude and Acceleration

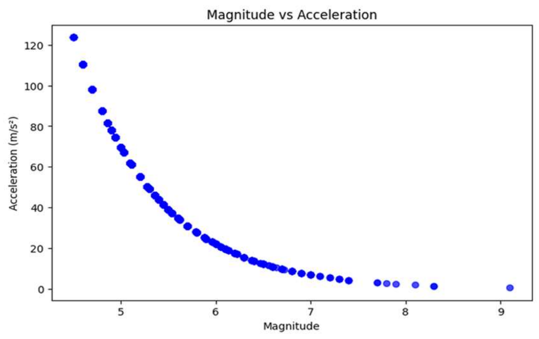

Fig 12. Magnitude vs Acceleration scatter plot

In Fig. 12, the scatter plot displays the relationship between earthquake magnitude and the resulting ground acceleration. Interestingly, the trend shows an inverse relationship, where higher magnitude earthquakes tend to produce lower acceleration values. This may be due to the wider energy dispersion in stronger earthquakes, which reduces the force experienced at specific points farther from the epicenter. In contrast, lower magnitude earthquakes often release energy more locally, resulting in higher localized acceleration. While most data follow this pattern, a few outliers suggest exceptions, potentially due to shallow depths or unique geological factors.

E. Relationship Between Earthquake Magnitude and Seismic Velocity

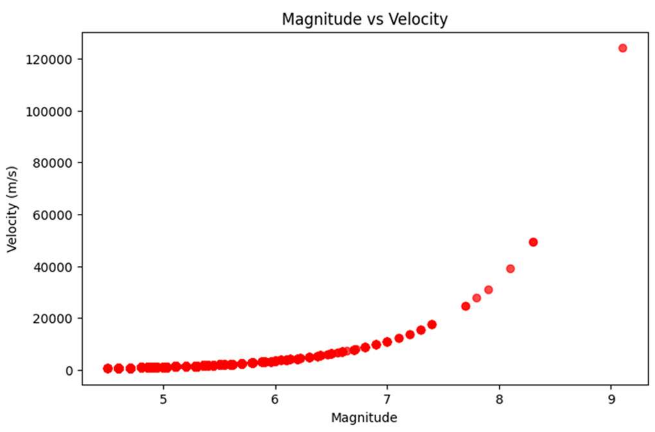

Fig 13. Magnitude vs Velocity scatter plot

In fig. 13, the scatter plot shows a clear trend as the magnitude of an earthquake increases, so does the speed of the seismic waves it produces. This makes sense because stronger earthquakes release more energy, causing the waves to travel faster through the ground. Most of the data supports this pattern, though there are a few exceptions, likely due to factors like how deep the quake occurred or the type of ground it moved through.

F. ACO Convergence

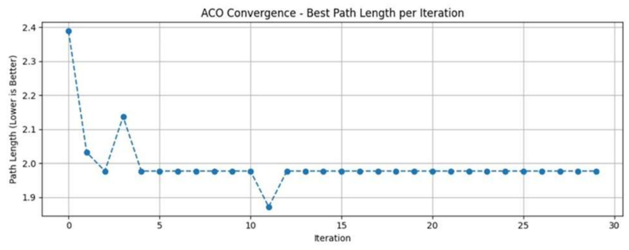

Fig 14. ACO Optimal Path

The Fig. 14 shows the convergence of ACO was evaluated by tracking the path length over iterations, representing the average distance of wave propagation paths found by the swarm. The key findings of ACO include:

- Initial Exploration (Iteration 0-1): High initial path length (~2.39) indicating random exploration of the solution space.
- Rapid Optimization (Iteration 1-2): A sharp decline in path length, suggesting early identification of efficient propagation routes.
- Stabilization Phase (Iteration 3-10): Minor fluctuations in path length (~1.96), signifying the algorithm's convergence to near-optimal solutions.
- Global Optimum (Iteration 11): Achieved a minimum path length (~1.87), possibly representing the most efficient seismic wave route.

G. XGBoost Results and Performance Evaluation

Metrics	Value
Learning Rate	0.1821
Mean Absolute Error	0.0002
Residual Range	±0.1
Prediction Deviation	Minimal (Most points very close to **y = x** line)

Table 2. XGBoost Metrics

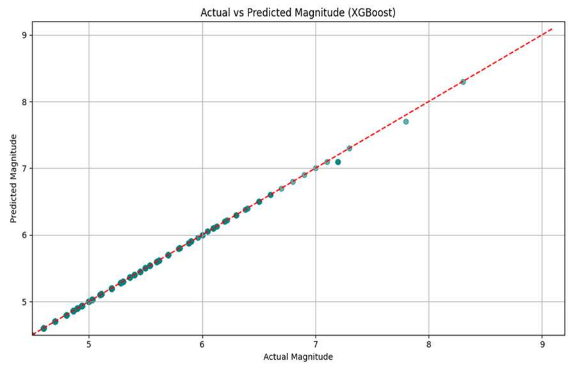

Fig 15. Visualization of the performance of the XGBoost

In Fig. 15, the scatter plot illustrates the performance of the XGBoost regression model in predicting earthquake magnitudes. Most of the predicted values closely align with the actual values. The concentration of points near the red dashed y = x line indicates high prediction accuracy. The minimal deviation between actual and predicted values represents a low error spread and supports the previously reported low Mean Absolute Error (MAE). Even at higher magnitudes above 8, the model maintains consistency, reflecting strong generalization across the entire magnitude range.

H. Residual Error Analysis of XGBoost Model Predictions

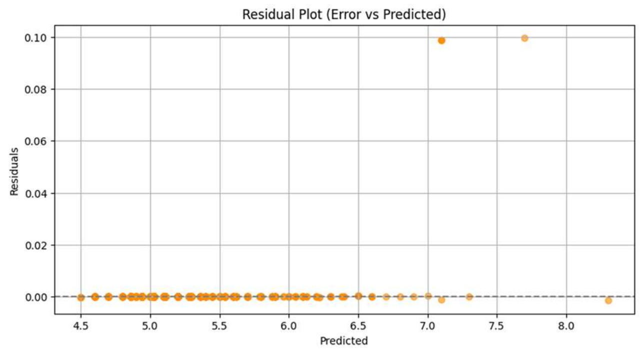

Fig 16. Residual Plot

In Fig. 16, the residual plot shows the difference between actual and predicted earthquake magnitudes using the XGBoost model. Most residuals are tightly clustered around zero, indicating high prediction accuracy and minimal deviation. The error magnitude is very low, rarely exceeding ±0.1, demonstrating the model's precision. The residuals are randomly scattered, suggesting no bias or underfitting. Only a few minor outliers are observed at higher predicted magnitudes.

I. Random Forest Parameters

Metrics	Precision	Recall	F1 Score	Support
High	1.00	1.00	1.00	59
Low	1.00	1.00	1.00	1307
Medium	1.00	1.00	1.00	1453
Accuracy	_	_	1.00	2819
Macro avg	1.00	1.00	1.00	2819

Weighed Avg	1.00	1.00	1.00	2819

Table 3. Performance Metrics of Random Forest Classifier

Table 3 shows the performance metrics of the Random Forest Classifier. It achieved perfect precision, recall, and F1 scores (all 1.00) for the three target classes: High, Low, and Medium. This indicates that the model was able to correctly identify every instance without any false positives or false negatives. The support column shows the distribution of samples in each class, with 59 samples labeled as High, 1307 as Low, and 1453 as Medium, totaling 2819 instances. Both the macro average and weighted average scores are also 1.00, showing that the model performed well.

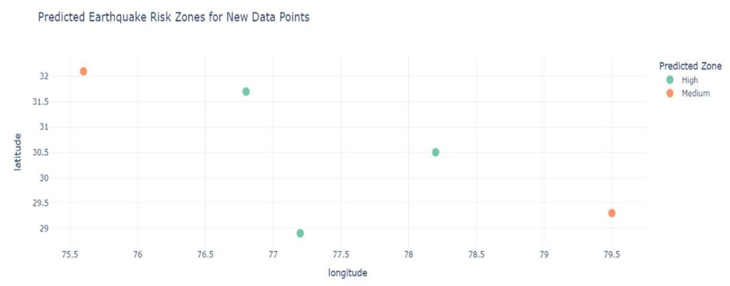

Fig 17. Predicted Earthquake Risk Zones for New Data Points

In Fig. 17, each point on the scatter plot represents a specific location for which the risk level was predicted using the classification model. The predicted risk zones are categorized as **High** (green) and **Medium** (orange), as shown in the legend.

V. CONCLUSION AND FUTURE WORK

In this study, we proposed a hybrid swarm intelligence framework for analysing seismic activity in Japan using machine learning techniques. By combining Particle Swarm Optimization (PSO) for feature selection and Ant Colony Optimization (ACO) for feature path optimization, we enhanced both the interpretability and predictive power of our models. The Kalman Filter effectively denoised seismic signals, making critical parameters such as magnitude, amplitude, and velocity more reliable for analysis. Isolation Forest and DBSCAN helped us uncover anomalies, while risk zoning categorized earthquake-prone regions into actionable levels of concern. Our PSO-tuned XGBoost model achieved highly accurate magnitude predictions with a mean absolute error as low as **0.0002**, and our Random Forest classifier achieved perfect accuracy in classifying seismic events into Low, Medium, or High risk zones. This integrated pipeline boosts the accuracy.

In the future, the model can be made even more realistic by incorporating directional heuristics, which considers the directionality of seismic wave propagation and fault line orientation. Rather than treating spatial features as static or isotropic, directional heuristics can help simulate how seismic energy travels through geological structures, prioritizing certain paths over others based on physical geography. This would enhance the predictive power of swarm algorithms like ACO, enabling them to follow more accurate paths. Incorporating such direction-sensitive logic could lead to more context-aware risk assessments, particularly in regions with known tectonic plate boundaries or fault lines.

VI. REFERENCES

[1] https://www.britannica.com/event/Japan-earthquake-and-tsunami-of-2011

[2] A Particle Swarm Optimization-Backpropagation (PSO-BP) Model for the Prediction of Earthquake in Japan Authors: Abey Abraham, V. Rohini Published in: Emerging Research in Computing, Information, Communication and Applications Publisher: Springer Singapore

[3] https://kamenpenkov.wordpress.com/wp-content/uploads/2016/01/handbook-of-swarm-intelligence-2011.pdf

[4]https://marcosoliveira.info/files/2017_OPMBM_exploration_exploitation_pso.pdf

[5] Particle swarm optimization Publisher: IEEE by J. Kennedy , R. Eberhart

[6] https://mat.uab.cat/~alseda/MasterOpt/ACO_Intro.pdf

[7] https://web2.qatar.cmu.edu/~gdicaro/15382-Spring18/additional/aco-book.pdf

[8]https://homes.luddy.indiana.edu/jbollen/I501F13/readings/dorigo99ant.pdf

[9] https://www.datacamp.com/tutorial/normalization-vs-standardization

[10] A Hybrid Quantum Particle Swarm Optimization based on Differential Evolution Strategy for Healthcare Applications 2025 AI-Driven Smart Healthcare for Society 5.0 | 979-8-3315-3633-6/25/$31.00 ©2025 IEEE | DOI: 10.1109/IEEECONF64992.2025.10963147 Shashank Mouli Satapathy, Hriday Agarwal , Abhyudoy Chaki ,Vidit Shah

[11] Multi-Agent With Multi Behavior Based on Particle Swarm Optimization (PSO) for Crowd Movement in Fire Evacuation Hartarto Junaedi , Mochamad Hariadi , I Ketut Eddy Purnama

[12] Time Series Analysis on Earthquakes Using EDA and ML DOI:10.1109/ICACSIS51025.2020.9263188 **Conference: 2020** International Conference on Advanced Computer Science and Information Systems (ICACSIS) Authors:Muhammad Fakhrillah, Fariz Darari ,Muhammad Rizqy Septyandy

[13] https://www.geeksforgeeks.org/xgboost/