IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 19, 2021, accepted July 29, 2021, date of publication August 9, 2021, date of current version September 10, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3103505

Earthquake Alert Device Using a Low-Cost Accelerometer and Its Services

SEONHYEONG KIM[1], IRSHAD KHAN[1], SEONHWA CHOI[2], AND YOUNG-WOO KWON[1]
[1]School of Computer Science and Engineering, Kyungpook National University, Daegu 41566, South Korea
[2]National Disaster Management Institute, Ulsan 44538, South Korea

Corresponding author: Young-Woo Kwon (ywkwon@knu.ac.kr)

This work was supported in part by Grant NDMI-2021-08-02-02, in part by the National Research Foundation of Korea (NRF), Korean Government (MSIT), under Grant 2021R1A5A1021944, in part by the NRF, Ministry of Education, through the Basic Science Research Program, under Grant NRF-2021R1I1A3043889, and in part by the National Disaster Management Research Institute under Grant 2021-08-02-02.

ABSTRACT With the recent increase in the number of earthquakes in Korea, research efforts have been directed toward the real-time detection of earthquakes and the formulation of evacuation plans. Traditional seismometers can precisely record earthquakes but are incapable of processing them on-site to initiate an alert and response mechanism. By contrast, internet of things (IoT) devices equipped with accelerometers and CPUs can record and detect earthquake signals in real time and send out alert messages to nearby users. However, the signals recorded on IoT devices are noisy because of two main factors: the urban buildings and structures these devices are installed in and their cost–quality trade-off. Therefore, in this work, we provide an effective mechanism to deal with the problem of false alarms in IoT devices. We test our previously proposed artificial neural network (ANN) with different feature window sizes ranging from 2 seconds to 6 seconds and with various earthquake intensities. We find that setting the size of the feature window to a certain interval (i.e., 4–5 seconds) can improve model performance. Moreover, an evacuation route guidance platform that considers user location is proposed. The proposed platform provides and visualizes information to user devices in real time through the communication between server and user devices. In the event of a disaster, safe shelters are selected on the basis of the information entered from the server, and pedestrian paths are provided. As a result, the direct and secondary damages caused by earthquakes can be avoided.

INDEX TERMS Earthquake, artificial neural network, low-cost MEMS sensor, evacuation.

I. INTRODUCTION

As the millions of earthquakes have been observed worldwide over the past 100 years, the resulting casualities are severe [1]. According to the average annual statistics for earthquake [2], an average of 27,000 people loses their lives every year and most injuries or deaths usually result from secondary events, such as the collapse of buildings and structures during evacuation [3]. Therefore, significant efforts have been exerted to accurately detect earthquakes in real time and provide appropriate action plans.

To detect earthquakes, a traditional approach calculates averages of accelerations for short and long time periods (i.e., STA/LTA) [4] and determines an earthquake if a sudden change occurs. Given its deterministic behavior and low computational cost, STA/LTA has been widely used in the last several decades. However, because of its straightforward detection mechanism, it is not suitable in noisy environments, in which STA/LTA's accuracy is greatly reduced because of low-quality data. Hence, recent research efforts have focused on incorporating machine learning approaches [5].

Internet of things (IoT) devices equipped with low-cost accelerometers have recently been proposed as tools for real-time earthquake detection [6]. MyShake is the first effort to involve smartphones as a seismic sensor. When a smartphone running an application detects an earthquake, the application forwards the data to the cloud for further processing to confirm an earthquake event [7]. In our prior work [8], [9], we introduced a standalone device that processes acceleration signals and alerts nearby users when an earthquake occurs. Citing the success of deep learning technologies, we adopted a lightweight convolutional neural network with a recurrent layer before the dense layer [10] to detect earthquakes among

The associate editor coordinating the review of this manuscript and approving it for publication was Mehdi Hosseinzadeh.

widely scattered sensors across the country and achieved 99% accuracy.

In this article, we introduce a newer version of the earthquake alert device consisting of two main modules for detecting earthquakes in real time and providing response services. The detection module detects earthquakes by using a state-of-the-art machine learning methodology, so that it can accurately discriminate earthquake data from non-earthquake data that include noise data that are generated from artificial or natural sources such as winds, human activities, etc.

In addition, in the event of an earthquake, people usually have difficulties in escaping dangerous areas because of the lack of information about evacuation routes or shelter locations, and also they have less perception of disaster risk [11]. Given the importance of quick evacuation to safe zones [12], evacuees need to be provided with the safety information including road hazards, the level of road and shelter congestion, and the presence of risk factors. Therefore, as soon as the earthquake alert device detects an earthquake, it broadcasts an alert through a broadcasting facility and sends out alert messages to nearby devices via multiple communication channels including Ethernet, Wi-Fi, and Bluetooh. Then, the nearby devices provide safety information based on the received alert message.

The contributions of this work are as follows:

- We developed an earthquake alert device that uses a low-cost accelerometer to monitor shakings and provides appropriate response services in case of earthquakes. To accurately detect an earthquake, we improved the machine-learning approach that we introduced in our prior work [9],
- We improved the earthquake detection model introduced in our prior work work [9] to reduce false alarms by increasing the size of a feature window. We first assessed how different sizes of the data input window affect the performances of multiple machine-learning based approaches which have recently been employed for earthquake detection using low-cost accelerometers [7], [9], [10]. As a result, the improved model achieved the accuracy from 94% to 97%.
- We implemented an evacuation guidance application that suggests nearby shelters based on a user's location in the event of an earthquake. The application provides real-time warnings about an earthquake event and information about evacuation routes to help users quickly evacuate to the nearest shelter.

This article is structured as follows. Section II describes the research efforts related to the preceding background technologies, and Section III describes the developed earthquake alert device and its software system. Section IV describes the evacuation guidance application. In Section V, we evaluate the developed earthquake alert device and its detection algorithm. Section VI presents the summary of this work.

II. RELATED WORK

The idea of earthquake early warning (EEW) was initiated in [13], in which they installed seismic sensors in California and once an earthquake is detected, they send a signal to nearby cities through telegraph. In the past few decades, EEW had been actively adopted in several countries [14]–[16] suffering from earthquakes and showed its usefulness and effectiveness. However, such systems are regional and use conventional high-quality seismometers. Moreover, the monitoring is performed by state-of-the-art conventional algorithms.

Conventional seismic methods involve computations and statistical observations to detect earthquakes in a waveform [17]. Many algorithms were proposed in the field of seismology but they are rarely used and only a few of them are applicable in real-time network's [18]–[20]. However, the precision of these methods is average and in a noisy environment it cannot be used as the main detection method.

Alternative approaches to traditional EEW have recently been developed on the basis of IoT devices equipped with low-cost accelerometers [8], [21]. These devices are installed in urban areas that can generate noise data because of not only sensor quality but also human involvement. Apart from such standalone approach, some systems leverage the increasing application of smartphones, e.g., MyShake [7]. In such approaches, traditional methods of thresholding are not sufficient to discriminate noise data from earthquake data; hence, machine learning approaches have attracted increasing attention because of their intelligent probabilistic behavior [22].

Recently, researchers cast earthquake detection as a supervised machine learning problem and proposed a variety of machine learning models. They take the advantage of deep learning methodology which has been successfully used in different domains [23], [24]. Deep convolutional models have been proposed to detect earthquake, location, P-wave, and determine the first motion polarity of an earthquake [25], [26]. In CRED [27], the authors proposed a 2D convolutional model where the deep layers contain bidirectional long short-term memory units to extract an earthquake segment from a 2D spectrogram of a 30-seconds earthquake waveform. Similarly, PhaseNet and EQTransformers [28], [29] have utilized the U-Net [30] architecture to segment P and S-wave from a raw waveform of a 30-seconds and 60-seconds respectively. These methods achieved very high performances, however, these are proposed for the offline earthquake detection problems and their applications in the IoT devices with constraint resources and time (the length of input window) are not possible. Moreover, these methods are designed for seismometer data of good quality, and devices are installed at secure sites with minimal human noise. In our prior work [10], we proposed the first lightweight convolutional recurrent neural network (CRNN) that operates on only a 2-second input waveform.

To realize earthquake detection by using low-cost MEMS sensors, in our prior work [9], we used a simple machine learning technique that takes the characteristics of earthquakes as a feature. Depending on operational environments,

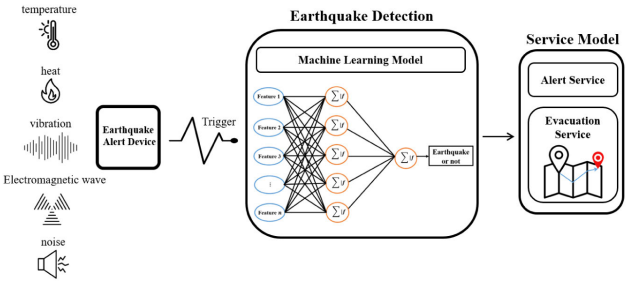


FIGURE 1. Approach overview.

different features and models were evaluated, and then we determined the most suitable machine learning-based model that can be used for both calm and noisy situations. However, there is relatively a high chance of false alarms when we use the model for low-cost MEMS sensors.

To find safe evacuation routes to destinations on the basis of user location after a disaster [31]. GPS data and accelerometer data from users' smartphones were collected to estimate safe paths and generate evacuation maps. In addition, the study did not use basic map data, and evacuation routes were identified on the basis of the safety of evacuation routes, evacuation distances, and evacuation times. However, the stability of evacuation routes remains limited in terms of route length and evacuate time, and the criteria for destinations are ambiguous. In addition, the safety about evacuation routes is only assessed, and no service is provided for users.

III. EARTHQUAKE ALERT DEVICE

In this section, we describe the development of the earthquake alert device including its hardware and software.

A. APPROACH OVERVIEW

Figure 1 shows the architecture of the proposed earthquake detection model.

When an alert device installed in a building detects noise, it triggers the earthquake detection model. Using the trained model, the device determines whether the noise data are seismic data in the earthquake detection module. When the input data is determined to be seismic data, then the service model provides two services. First, the alert service can be provided through built-in speakers by sending information to the alert device installed in the building. Second, the evacuation service also sends information about the evacuation route to the user's mobile device, thereby allowing the user to safely escape dangerous areas and evacuate to a safe shelter.

B. HARDWARE SYSTEM DEVELOPMENT

The developed earthquake alert device is manufactured using an integrated PCB and a Raspberry compute module. It also has a smoke detection sensor for fire alarms and a speaker for broadcasting. Figure 2 shows the developed device and its external ports including RS-485, Ethernet, USB, and audio ports. The system can be connected directly to a broadcasting facility installed in a building through the audio port or broadcast an earthquake alarm within a limited area using the built-in speaker.

FIGURE 2. Earthquake alert device.

The earthquake alert device is easily affected by the surrounding environment, including the operating temperature, electromagnetic waves, heat, noise, and vibration. In the following discussion, we describes the effects of the surrounding environment to the earthquake alert device.

- **Operating temperature:** Low temperatures can cause problems such as delays in data processing or power cuts. By contrast, high temperatures can affect the MCU or other equipment. Therefore, installation guidelines should specify the need to avoid cooking facilities, heaters, and direct sunlight when installing the earthquake alert system.
- **Electromagnetic waves:** Although the earthquake alert system can receive frequency interferences from nearby radio communication devices, the effects of the alarm due to electromagnetic waves are very limited because radio communication functions are not used.
- **Heat:** The heating test using stressberry [1] shows that the stress situation is above 73 °C. Furthermore, the temperature outside the case is affected by the external temperature rather than the temperature of the board.
- **Noise:** Noise can degrade the quality of the acceleration sensor, and it can be detected by the acceleration sensor because of the large speakers. Considering this factor, the installation location of the earthquake alert device must be decided carefully.
- **Vibration:** The most significant factor affecting the ability of the device to detect earthquakes is a sudden spike of acceleration occurring at the hardware level of the device. The effects of external vibration exert a significant impact on the seismic detection performance of the earthquake alert device.

[1] https://pypi.org/project/stressberry/

TABLE 1. CRNN model architecture for earthquake detection.

Layer	Stage	No. of filters	Activation	Filter size
1	C	64	ReLU	3
2	C	64	ReLU	3
3	Max-Pooling	-	-	-
4	Flatten	-	-	-
5	RNN	100	TanH	-
6	F	100	ReLU	-
7	Output	1	sigmoid	-

C. MACHINE LEARNING MODELS FOR EARTHQUAKE DETECTION

We considered three machine learning models for detecting real-time earthquakes by using low-cost accelerometers. Two of them are based on the traditional ANN models and comprise three and five input features. The third model uses a deep neural network that combines convolutional neural network and recurrent neural network (RNN) models. Below is the complete description of each model.

1) ARTIFICIAL NEURAL NETWORK WITH THREE INPUTS (ANN3)

The first model for evaluation is MyShake [7], as it is the first global real-time earthquake detection algorithm to utilize low-cost accelerometer sensors. The model architecture of MyShake is shown in Figure 3a. The model has three features: interquartile range (IQR), zero crossing (ZC) rate, and cumulative absolute velocity (CAV). IQR and CAV constitute the amplitude feature while ZC is the frequency feature. Herein, IQR is the mid 50% of the vector sum of three seismographic components, CAV is the cumulative measure of the vector sum for the given time window, and ZC is the maximum zero crossing rate of the three components.

2) ARTIFICIAL NEURAL NETWORK WITH FIVE INPUTS (ANN5)

The second ANN that we selected for evaluation was that proposed in [9]. This model is similar to MyShake, but it splits the frequency component into three separate features, namely, Max ZC, Min ZC, and Max Non-ZC. Max ZC counts the zero crossings of the maximum acceleration component in the case of zero crossing at multiple components at the same time; for single-component zero crossings, it is identical to ANN3's ZC feature. Conversely, Min ZC counts the zero crossings of the minimum acceleration component. Max Non-ZC counts the frequency of maximum acceleration in the absence of zero crossing. The model architecture is given in Figure 3b.

3) CONVOLUTIONAL RECURRENT NEURAL NETWORK (CRNN)

We also evaluated the deep neural model used in CrowdQuake proposed in [10]. The model uses a 1D CNN with an additional RNN layer before the dense layer.

The model architecture is given in Table 1, where C denotes the convolution and F denotes the fully connected layer. The CRNN uses a recurrent unit in which the previous output is fed back with the current input. For example, a 2 s input sequence is divided into two subsamples of 1 s each. These two subsamples are fed sequentially into the model; the output of the first subsample is fed back through the model along with the second subsample.

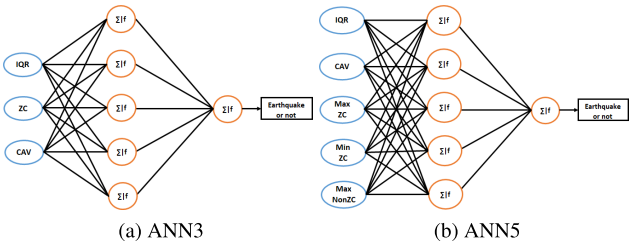


FIGURE 3. Earthquake detection models: ANN3 and ANN5.

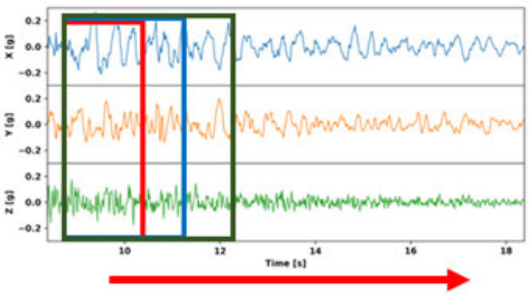


FIGURE 4. Window-based earthquake detection.

4) SLIDING WINDOW-BASED REAL-TIME DETECTION

To further increase the possibility of seismic detection, we classified the data into two groups on the basis of 4.0 magnitude. At this point, different models were intended to be applied depending on whether the maximum acceleration value of the event exceeded the 4.0 magnitude level when the event surpassed the trigger criteria. The performance was then measured by increasing the window size for learning from 2-second to 6-second data. To evaluate the performance of the proposed model, we compared it with the ANN3 and ANN5 models and the CRNN models based on deep learning.

The proposed model performed better on the 6-second data than on the 2-second data. Therefore, this study adopted a method of learning that uses a 2-second window simultaneously with 2-, 3-, and 4-second windows for earthquake detection. Moreover, the method detects earthquakes by moving each window for 1-second at a time.

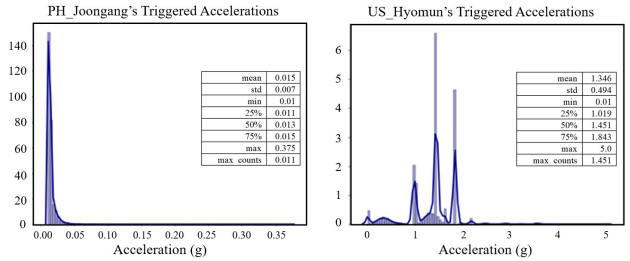
Figure 4 schematizes the use of different window sizes. As seismic detection lasted up to 10 seconds, the 2-, 3-, and 4-second windows are run 9, 8, 7 times, respectively. The computational time is acceptable for the real-time processing.

D. DATA MANAGEMENT

The earthquake alert device periodically transmits status information and data together with the event information to a management server when an event occurs. At present, when events due to earthquakes and motions exceeding a certain

```
FullEventMetaData  {
    publisher_uuid*   string
                      example: a2662358-276a-4785-933a-8c58a6e2b2e0
                      pattern: [0-9a-fA-F]{8}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{12}
    recorded*         string($date-time)
    verified*         boolean
    id*               string
    published*        string($date-time)
    file_posted*      boolean
}
```

FIGURE 5. Metadata for storing events for seismic alarms.



(a) normal devices (b) abnormal devices

FIGURE 6. The PGA distribution: normal devices and abnormal devices.

threshold (0.002 g), and acceleration data are sent for post-analysis.

When an earthquake occurs, the earthquake alert device sends the recorded acceleration data to a proxy server, which then forwards the data to the data processing server. The event data sent to the data processing server are then stored in the database after frequency component analysis. Furthermore, the results of the frequency component analysis are graphically represented and stored as an image file for retrieval. The event and acceleration data are stored in the database, with the event data viewable by the administrator through a web server.

Figure 5 shows the metadata of the acceleration data for storing events recorded by the earthquake alert system. The database consists of two types: event and seismic activity data stored together and device information. The types of data stored in the database consist of acceleration instrumentation data and metadata for earthquake alerts. The acceleration instrumentation data are divided into event data and seismic data. Event data include all vibrations over a century applied to the earthquake alert system, and seismic data include the seismic behavior by seismic detection algorithms.

E. TESTBED

For the testbed, we installed the developed devices at 29 public service offices located in three different cities. The devices are connected to the management server via the internet, and status messages and important events are sent to the management server for further analysis.

The collected data are used to analyze the environmental characteristics of the installed location and issued alarms. Noise data or non-earthquake event data are used for the model training. The event data are analyzed to determine a threshold triggering an earthquake detection model. Figure 6 shows the peak ground acceleration (PGA) distribution plot of events collected from a normal device. Initially, we set the trigger threshold to 0.001 g, however, the device reported too many triggers in a short time period. Thus, after analyzing the noise levels of the installed devices, we set the trigger threshold to 0.0025 g to reduce the number of seismic data records. Because devices has different operational environments, we set different thresholds for them after analyzing all the triggered events.

IV. EARTHQUAKE EVACUATION SERVICE

The proposed evacuation service visualizes various information received from server devices on user devices. In the event of an earthquake, the location of a nearby shelter is indicated on the map, centering on the user's location. When the user is in a specific area or market, the exit of the zone is received from the server device and selected as a stopover. In addition, the data received from the alert device [9] and the information received from server devices are used to determine the safety of specific areas. After determining the final destination on the basis of the safety of the shelter determined, the route to the shelter is provided as a pedestrian route.

A. EVACUATION ROUTE GUIDANCE PLATFORM

The proposed platform provides information to users in real time and visualizes it. Stored information about nearby areas of a user's location is generally provided. In the event of a disaster, the safest shelter is selected, and pedestrian paths are provided using the distance between the user and the shelter and the attribute values received from the server. In addition, the safety and capacity of the shelter are determined in real time to rediscover the user's evacuation route.

B. EARTHQUAKE RESPONSE FRAMEWORK

Figure 7 shows the framework for earthquake response. It describes the framework in which server devices provide information, and user devices receive information after disasters such as earthquakes. Through P2P connections to Wi-Fi Direct, user devices can be located, and messages can be sent within the specified radius of the server device. In addition, user devices can receive messages from server devices to the cloud regardless of the distance between devices. In the event of an earthquake, information about dangerous areas and about exits based on user location is automatically downloaded and visualized on user devices. The message is delivered in JSON format.

C. RECOMMENDATION OF EVACUATION PATHS

The proposed evacuation route searching algorithm is described below.

Input includes location $user_{loc}$ of the user received via GPS and location $safe_i$ of the shelter. Safety is determined according to the degree of danger of the shelter. Maximum capacity $count^{max}$ and current capacity $count^{now}$ are also measured for safe shelters. Therefore, the shelter that is closest to the user's location and has a large capacity for evacuees is selected. The

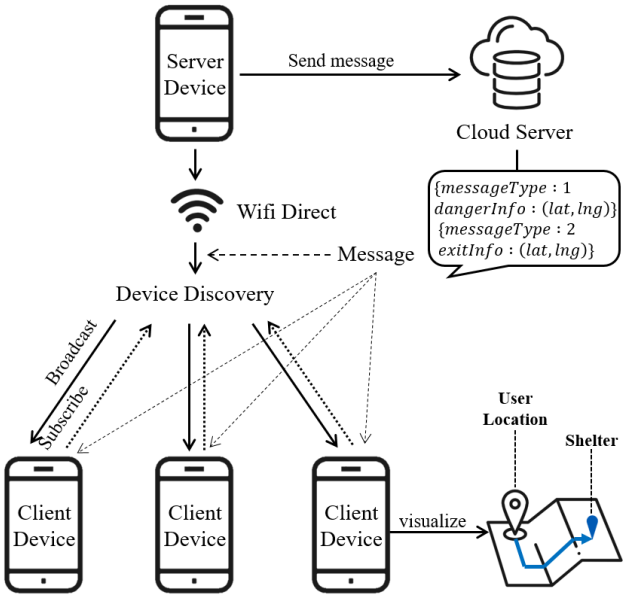


FIGURE 7. A framework for earthquake response services.

Algorithm 1 Algorithm for Searching an Evacuation Path

Input: $user_{loc}$ is the location of device user, $safe_i$ is the shelter)
Output: *path* is the pedestrian route to shelter

if $recog(user_{loc}) = true$ **then**
  $SAFE[] \leftarrow Min(dis(User_{loc}, shel[lat][lng]), ASC)$
  if $intensity(safe_i) > \alpha || server(safe_j)$ **then**
    $SAFE[i] = NULL$
    $SAFE[j] = NULL$
  end if
  for each $safe_i$ in *SAFE* **do**
    update()
    $capacity[] \leftarrow (count^{max} - count^{now})$
  end for
  $final \leftarrow destination(SAFE[i]^{min}, capacity[]^{max})$
  $path \leftarrow navigate(User_{loc})$
end if
return *path*

information on shelters is updated and reflected in real time so that users can choose from more than one shelter.

Instead of simply providing a route to a nearby shelter, the algorithm compares the earthquake measurement data received from the alert device with the hazard level threshold to determine the safety of the shelter. It also aims to provide information on dangerous shelters from server devices, excludes such locations from the search, and provides routes to safe shelters.

D. IMPLEMENTATION OF THE EVACUATION SERVICE

Figure 8 shows the implementation of the scenario assuming that the user is on a certain street in a market and that an earthquake is occurred. The left figure provides a pedestrian

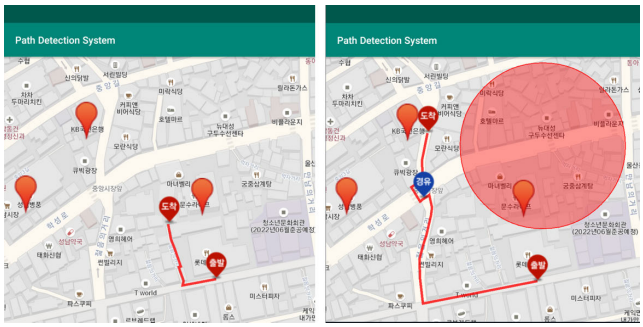


FIGURE 8. Scenario implementation.

evacuation route to the shelter near to the user's location. The alarm broadcasting from the terminal equipment installed in the customer center in the market determines the dangerous area in advance and enters the degree of danger of the shelter in the system. The right figure presents an evacuation route to a safe shelter provided by the server device, along with location information for the hazardous area and exit location. Hazardous zones are marked with a circle, and shelters within the zones are excluded from the search. The nearest shelter based on the user's location is set as the destination.

The proposed system determines the safest shelter depending on the user's location in the event of an earthquake or disaster. It also provides a pedestrian evacuation route that reflects real-time data so that users can easily move to shelters. The system marks the location of shelters and a user collected through GPS on the map. The locations of all shelters within the scope of the user's location (special city, metropolitan city, city) are also provided using the National Civil Defense Evacuation Facilities[2]' standard data. A pedestrian evacuation route is generated by determining the safety of shelters in real time.

V. EVALUATION

In this section, we evaluate our earthquake detection models using a deep neural network and two artificial neural networks using different training methods. We comprehensively explain the experimental procedures used for the models under review.

A. DATASET

We use two datasets in our experiments;

- Earthquake data downloaded from the National Research Institute of Earth Science and Disaster Prevention (NIED) [32]. We downloaded seismograms from May 1996 to March 2020 and obtained 14,145 earthquake records. All the data were normalized (subtracting mean) and converted from counts to unit g (9.8 m/s^2). Deep net models can be trained on an entire dataset, whereas traditional models are limited and can be trained only on a certain range of intensities. Therefore, we cate-

[2]Shelter data was retrieved from www.safekorea.go.kr on Oct. 2020.

TABLE 2. Dataset for the evaluation.

Category	Number of records
Low (PGA > 0.05g)	4,795
High(PGA < 0.05)	9,350

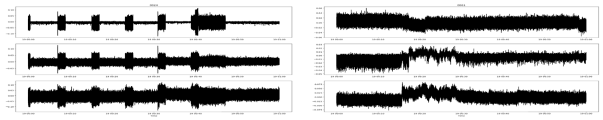


FIGURE 9. Sensors' noise.

gorized the earthquake data into low- and high-intensity data. Low-intensity data included earthquakes whose X-component (EW) PGA is less than 0.05 g. For high-intensity earthquakes, we selected earthquakes whose PGA values are greater than or equal to 0.05 g. Table 2 shows the number of earthquakes in each resulting category.

- This dataset includes recordings collected from deployed stationary sensors and human activity data recorded on smartphones. Low-cost accelerometer sensors are usually installed at sites where noise can be introduced by anthropogenic or natural sources; moreover, the sensors can themselves introduce noise. Therefore, we included some human activity and noise data as a non-earthquake dataset. Human activity data were recorded on mobile phones for several hours. The dataset includes recordings of activities, such as bus riding and staying, indoor and outdoor noise data, and recordings of shaking desks. We included 1 hour of data from each of the 20 sensors with different noise patterns (Figure 9).

B. MODEL EVALUATION

1) PRE-PROCESSING

The data recorded on low-cost sensors usually have intrinsic noise; we therefore applied a bandpass filter of 0.1–10 Hz to all the data used. For the traditional machine learning models, we further applied the standard scaler normalization to the scaled feature sets to speed up the training process and increase the performance. Then, to transform the data collected by the NIED and PEER into data collected by the sensor environment, we reduced the sampling rate to 50 Hz and inserted the noise collected by the sensor. The data collected and the existing seismic data were used together for learning. By reducing the proportion of human movement data, earthquake-induced shaking was sought from the building's constant/emergency vibration data. We also intended to improve accuracy by learning different models depending on earthquake magnitude. Figure 10 shows the results of the addition of the noise data collected by the sensors to the seismic acceleration values observed at high-quality stations.

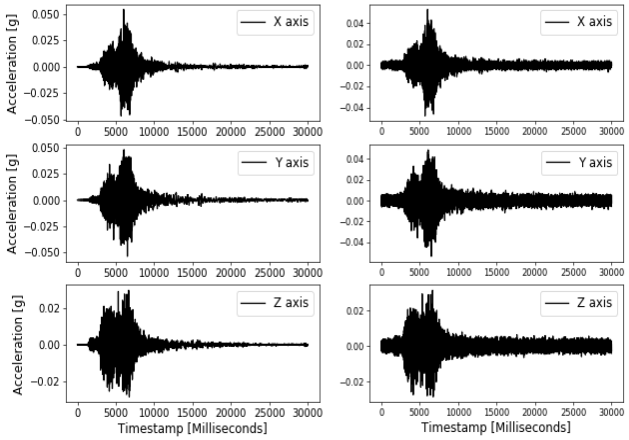


FIGURE 10. Results of data processing by adding noise.

2) MODEL TRAINING

The traditional models were trained on a feature set extracted from the raw data (X, Y, and Z axis). The deep net model requires no simplified feature set; it can operate on the raw data directly. We split the raw data into 70% and 30% for training and testing, respectively.

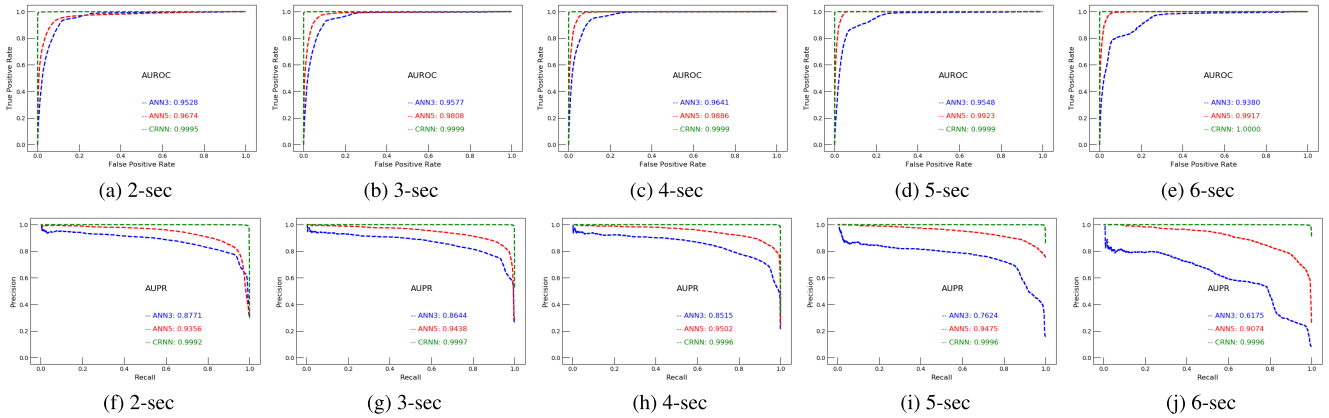
We trained ANN3 and ANN5 models on 70% of the earthquake data and an equal number of centroids of the non-earthquake data because the dataset was imbalanced. The centroids were calculated using the k-means algorithm [33], where c is equal to the number of earthquake data points used for training. The non-earthquake data was grouped into c clusters. For testing, we used the original 30% test data of the non-earthquake class instead of the centroids so that the centroids were only calculated for the training data. However, because the deep net model's input was simply the raw data, we resolved the data imbalance issue by computing the following weighted cross-entropy loss function l:

$$l = -[w_0(1 - y_t)\ln(1 - \hat{y_t}) + w_1 y_t \ln \hat{y_t}], \tag{1}$$

where $\hat{y_t}$ is the true class label of the input data, and w_0 and w_1 are the weights for the non-earthquake and earthquake classes, respectively. To resolve the data imbalance problem, $w_0 = 1$ and w_1 to be the ratio of non-earthquake data to the volume of the earthquake data such that the total cumulative weights for both classes were the same as that used in [10]. We use the same 70% and 30% split of raw data for training and testing the deep net model, respectively.

3) EXPERIMENTAL SETUP

In the literature, sampling rates of 25, 50, and 100 Hz have been used to train and test machine learning models. We conducted exhaustive experiments and found that the performance of all of these models was relatively unaffected by varying sampling rates; for this work, we arbitrarily chose the 50 Hz sampling rate because it is the median sampling rate found in the literature.



(a) 2-sec (b) 3-sec (c) 4-sec (d) 5-sec (e) 6-sec

(f) 2-sec (g) 3-sec (h) 4-sec (i) 5-sec (j) 6-sec

FIGURE 11. AUROC (Top row) and AUPR (Bottom row) of 6-second earthquake window with 2-, 3-, 4-, 5-, and 6-second feature-window (left to right) configurations of the models trained with high intensity earthquake data.

The ANN models were implemented in Python SKlearn [34], and the deep net model was implemented in Keras libraries [35]. For the ANN models, we used the stochastic gradient descent optimization function with the adaptive learning rate of 0.2 for updating the weights [36]–[38]. To avoid overfitting, we used an early stopping strategy [39]. The sigmoid activation function that we used for the hidden and output layers is defined as a function f on input x as;

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2}$$

We used the same configuration proposed in [10] for the deep net model (i.e., backpropagation [40] and mini-batch gradient descent [41] with the size of 256 and with an Adam optimizer [42]) to optimize the model parameters and weights. The training process took 100 epochs, and dropout was used [43] with a probability of 0.5 at the fully connected layer.

4) PERFORMANCE MEASUREMENT

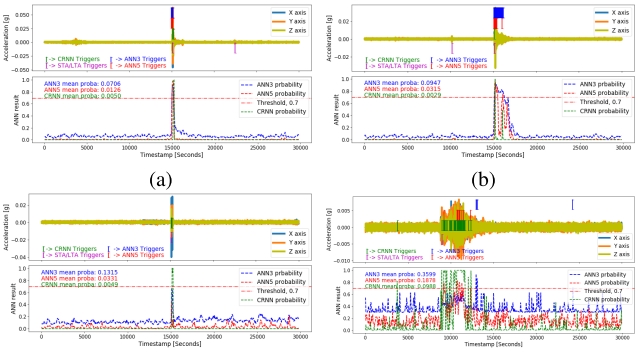
We calculated the accuracy (Acc), precision (Pre), and recall (Rec) from the performance matrices recording true positive (TP), false positive (FP), true negative (TN), and false negative (FN) results. Accuracy indicates the overall performance of the model, precision indicates how accurately the model predicts earthquakes, and recall denotes model's sensitivity. We also showed the receiver operating characteristic (ROC) curves and precision–recall (PR) curves for all test cases to show the trade-off between the TP rate and FP rate and between precision and recall, respectively.

C. RESULTS AND DISCUSSION

We selected the strongest 6-second and 8-second portions of each earthquake waveform (i.e., earthquake window) and extracted 2, 3, 4, 5, and 6-second feature windows for training and testing the models. A threshold value of 0.7 was set for all experiments. The results of the models trained with high- and low-intensity data are shown in Tables 3 and 4,

TABLE 3. Summary of the models' test performance on 30% data, trained with high-intensity earthquake data.

Setting	Model	TP	FP	TN	FN	Acc.	Pre.	Recall
6S-2S	ANN-3	10,222	1,762	31,025	3,793	88.13	85.30	72.94
	ANN-5	10,874	1,003	31,784	3,141	91.15	91.56	77.59
	CRNN	13,780	46	32,741	235	99.40	99.67	98.32
6S-3S	ANN-3	7,374	1,201	31,562	3841	88.54	85.99	65.75
	ANN-5	9,599	1,104	31,659	1616	93.82	89.69	85.59
	CRNN	11,142	50	32,713	73	99.72	99.55	99.35
6S-4S	ANN-3	5,556	987	31,752	2854	90.67	84.92	66.06
	ANN-5	7,441	879	31,860	969	95.51	89.44	88.48
	CRNN	8,335	29	32,710	75	99.75	99.65	99.11
6S-5S	ANN-3	3,867	1,187	31,528	1743	92.35	76.51	68.93
	ANN-5	5,074	630	32,085	536	96.96	88.96	90.45
	CRNN	5,594	115	32,600	16	99.66	97.99	99.71
6S-6S	ANN-3	2,052	1,626	31,066	753	93.30.	55.79	73.16
	ANN-5	2,477	654	32,038	328	97.23	79.11	88.31
	CRNN	2,792	34	32,658	13	99.87	98.80	99.54



(a) (b) (c) (d)

FIGURE 12. Prediction results of the models (a-c) Models trained with high intensity earthquake dataset (d) Model trained with low intensity earthquake dataset.

respectively. In the Setting column, 6s-2s means that the model was trained with the strongest 6 second portion of an earthquake and that a 2-second feature window was used. We tested five feature windows per earthquake sample. All three models showed very good performance on the 30% test

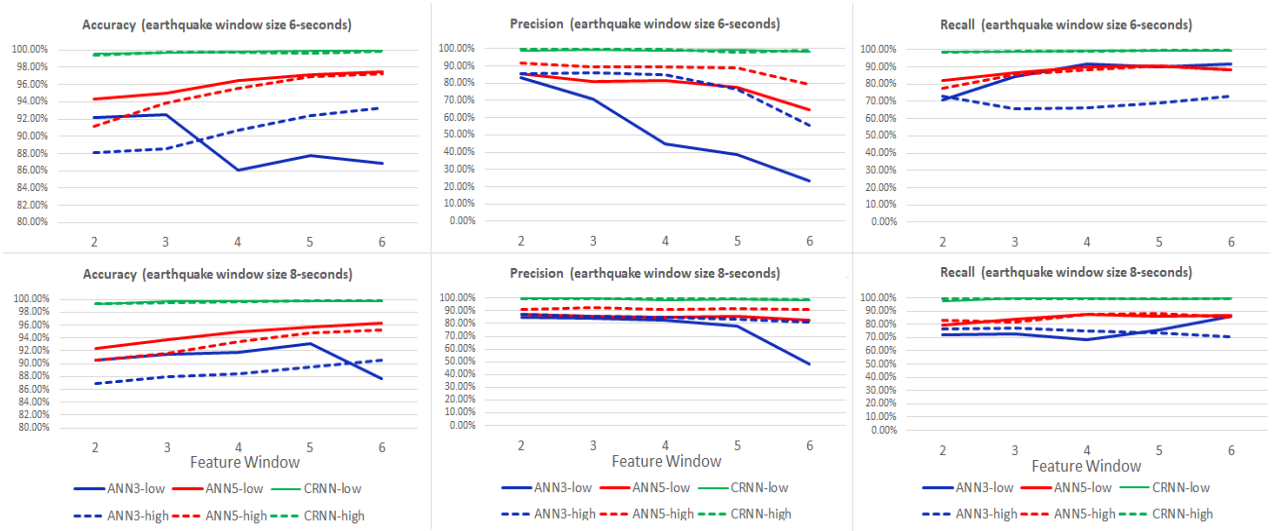


FIGURE 13. Graphical comparison of the models' accuracy, precision, and recall with increasing in feature-window size.

TABLE 4. Summary of the models test performance on 30% data trained with low-intensity earthquake data.

Setting	Model	TP	FP	TN	FN	Accu:	Pre:	Recall
6S-2S	ANN3	5,111	1,037	31,750	2,079	92.21	83.13	71.08
	ANN5	5,909	1,012	31,775	1,281	94.26	85.38	82.18
	CRNN	7,111	73	32,714	79	99.62	98.98	98.90
6S-3S	ANN3	4,853	1,998	30,765	902	92.47	70.84	84.33
	ANN5	4,973	1,146	31,617	782	94.99	81.27	86.41
	CRNN	5,670	32	32,731	85	99.70	99.44	98.52
6S-4S	ANN3	3,947	4802	27,937	368	86.05	45.11	91.47
	ANN5	3,892	890	31,849	423	96.46	81.39	90.20
	CRNN	4,299	62	32,677	16	99.79	98.58	99.63
6S-5S	ANN3	2,591	4,071	28,644	289	87.75	38.89	89.97
	ANN5	2,602	743	31,972	278	97.13	77.79	90.35
	CRNN	2,855	25	32,690	25	99.86	99.13	99.13
6S-6S	ANN3	1,321	4,377	28,315	119	86.83	23.18	91.74
	ANN5	1,275	704	31,988	165	97.45	64.43	88.54
	CRNN	1,429	22	32,670	11	99.90	98.48	99.24

TABLE 5. Models' comparison with STA/LTA.

method	Low Intensity			High Intensity		
	TP	FN	Recall	TP	FN	Recall
ANN3	1336	102	92.91	1874	930	66.83
ANN5	1303	135	90.61	2415	389	86.13
CRNN	1422	16	98.89	2782	22	99.22
STA/LTA	947	491	65.86	1710	1094	60.98

data. The CRNN model outperformed the ANN models for all metrics. ANN5 performed better than ANN3, particularly as the feature window size increased. The 6-second earthquake window size was suitable for the ANN models. On the basis of the results shown in the tables, we ranked the models in the following order: CRNN, ANN5, and ANN3.

The AUROC and AUPR of the 6-second earthquake windows (Figure 11) clearly demonstrate the same ranking of models. CRNN showed almost the same results in every case and was invariably the best among the three models. The other two models, however, showed variable results; for instance, ANN5 showed an increase in performance when we increased the feature window size, with 4- and 5-second appearing to be the optimal feature window settings for the model. For ANN3, AUROC increased when the feature window size increased, whereas AUPR decreased for the 5 and 6-second windows (as shown in Figures 11i,11j). As a result, the best configuration in terms of AUROC and AUPR was determined to be the 4-second window.

Figure 13 displays graphically the overall test accuracy, precision, and recall status with respect to the changes in the feature window sizes of the models. The accuracy and recall of ANN5 increased as the size of the feature window increased, and its precision remained approximately the same. By contrast, ANN3 shows a relatively low performance, especially when trained with low-scale earthquakes.

To compare these machine learning models with traditional seismic methods, we used recursive STA/LTA to detect the maximum acceleration of an earthquake in each of the low- and high-intensity earthquake datasets. Setting the time and threshold parameters of the STA/LTA is quite difficult; we used the default setting, due to its optimal results in the experiments (i.e., 5/10 and 1.5 threshold values). We counted TP (detected) when the maximum acceleration of an earthquake sample was within the 5-second range of the STA/LTA trigger; otherwise, FN (missed) were counted. We trained the machine learning models on the 2-second earthquake window with the same length of feature-window, and also on non-earthquake datasets. In this way, the model's TP or FN shows only whether the model detected or missed the maximum acceleration of an earthquake. Table 5 compares the machine learning recall rate with the STA/LTA recall rate. The results of the machine learning models are an improvement over those of the STA/LTA method.

1) PREDICTION

We tested the models and the traditional triggering method (STA/LTA) on the data of real earthquakes that recently struck South Korea, as recorded on low-cost accelerometers. All models trained on the high-intensity earthquake dataset accurately detected the earthquake with a PGA of.06 (Figure 12a). When we tested these models on low-intensity data (e.g., PGA of.03), the CRNN model was able to detect more low magnitude earthquakes than ANN3 and ANN5 (Figure 12c). The result was expected because traditional models are trained with two amplitude features and these models can be affected by earthquake intensity. If we trained the models on the low-intensity earthquake dataset, then they would be able to detect very low-amplitude earthquakes (Figure 12d) but with high probabilities of false alarms. This case is especially true for ANN3 and ANN5; the mean probabilities of these two models are higher than that of the CRNN, with ANN3 having the highest among the three. STA/LTA triggered accurately but with a high probability of false alarms because of fixed threshold value (Figure 12a and 12b).

2) PARAMETERS

The parameters of a model are the numbers of trainable parameters (i.e., weights and biases). The numbers of trainable parameters of CRNN, ANN5, and ANN3 are 340,493, 36, and 26, respectively. Given its deep architecture, the CRNN comprises a very large number of parameters and thus requires more memory and computational power than traditional models such as ANN5.

VI. CONCLUSION

In this article, we aim to develop an earthquake alert device and its services as well as improving the performance of the developed device. Using more earthquake dataset and different window sizes, we tested recently proposed earthquake detection models used for IoT devices. As we increase the window size, all models show promising results, with CRNN demonstrating the best performance, followed by ANN5 and ANN3. However, we believe that the deep neural network is not necessarily the best choice for the earthquake alert device because of the computational cost. As a result, we use the ANN5 with variable window sizes ranging from 2 seconds to 4 seconds.

In addition, we implemented an evacuation route guidance service that provides useful safety information before an earthquake and the locations of shelters or other safe places are suggested after an earthquake. The service uses visual or social data to determine dangerous areas that should be avoided and then the best route is suggested. In the future research, we will provide various safety-related services tailored to specific disaster situations.

REFERENCES

[1] S. Wiemer, *Earthquake Statistics and Earthquake Prediction Research*. Zürich, Switzerland: Institute of Geophysics, 2000.

[2] *Worldwide Average Annual Statistics for Earthquake Counts*. [Online]. Available: https://www.usgs.gov/natural-hazards/earthquake-hazards/lists-maps-and-statistics

[3] G. Bernardini, S. Santarelli, E. Quagliarini, and M. D'Orazio, "Dynamic guidance tool for a safer earthquake pedestrian evacuation in urban systems," *Comput., Environ. Urban Syst.*, vol. 65, pp. 150–161, Sep. 2017.

[4] A. Trnkoczy, "Understanding and parameter setting of STA/LTA trigger algorithm," in *Proc. New Manual Seismol. Observatory Pract. (NMSOP)*, 2009, pp. 1–20.

[5] M.-A. Meier, Z. E. Ross, A. Ramachandran, A. Balakrishna, S. Nair, P. Kundzicz, Z. Li, J. Andrews, E. Hauksson, and Y. Yue, "Reliable real-time seismic signal/noise discrimination with machine learning," *J. Geophys. Res.*, vol. 124, pp. 788–800, Jan. 2019.

[6] A. Alphonsa and G. Ravi, "Earthquake early warning system by IOT using wireless sensor networks," in *Proc. Int. Conf. Wireless Commun., Signal Process. Netw. (WiSPNET)*, Mar. 2016, pp. 1201–1205.

[7] Q. Kong, R. M. Allen, L. Schreier, and Y.-W. Kwon, "MyShake: A smartphone seismic network for earthquake early warning and beyond," *Sci. Adv.*, vol. 2, no. 2, Feb. 2016, Art. no. e1501055.

[8] J. Lee, I. Khan, S. Choi, and Y.-W. Kwon, "A smart iot device for detecting and responding to earthquakes," *Electronics*, vol. 8, no. 12, p. 1546, Dec. 2019.

[9] I. Khan, S. Choi, and Y.-W. Kwon, "Earthquake detection in a static and dynamic environment using supervised machine learning and a novel feature extraction method," *Sensors*, vol. 20, no. 3, p. 800, Feb. 2020.

[10] X. Huang, J. Lee, Y.-W. Kwon, and C.-H. Lee, "CrowdQuake: A networked system of low-cost sensors for earthquake detection via deep learning," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. New York, NY, USA: Association for Computing Machinery, 2020, pp. 3261–3271, doi: 10.1145/3394486.3403378.

[11] Y. Ao, K. Huang, Y. Wang, Q. Wang, and I. Martek, "Influence of built environment and risk perception on seismic evacuation behavior: Evidence from rural areas affected by Wenchuan earthquake," *Int. J. Disaster Risk Reduction*, vol. 46, Jun. 2020, Art. no. 101504.

[12] B. Y. Heo and S. H. Park, "A study on improvement of the procedure of selection and maintenance for natural disaster shelter," *Korean Soc. Hazard Mitigation*, vol. 16, no. 5, pp. 105–112, Aug. 2016.

[13] Y. Nakamura and B. E. Tucker, *Japan's Earthquake Warning System: Should it be Imported to California*. Emmitsburg, MD, USA: National Emergency Training Center, 1988.

[14] D. D. Given, R. M. Allen, A. S. Baltay, P. Bodin, E. S. Cochran, K. Creager, R. M. de Groot, L. S. Gee, E. Hauksson, T. H. Heaton, M. Hellweg, J. R. Murray, V. I. Thomas, D. Toomey, and T. S. Yelin, "Revised technical implementation plan for the ShakeAlert system—An earthquake early warning system for the West Coast of the United States," US Geological Survey, Reston, VA, USA, Tech. Rep., 2018.

[15] M. Hoshiba, O. Kamigaichi, M. Saito, S. Tsukada, and N. Hamada, "Earthquake early warning starts nationwide in Japan," *EOS, Trans. Amer. Geophys. Union*, vol. 89, no. 8, pp. 73–74, 2008.

[16] H. Alcik, O. Ozel, N. Apaydin, and M. Erdik, "A study on warning algorithms for Istanbul earthquake early warning system," *Geophys. Res. Lett.*, vol. 36, no. 4, 2009, Art. no. L00B05.

[17] W. F. Freiberger, "An approximate method in signal detection," *Quart. Appl. Math.*, vol. 20, no. 4, pp. 373–378, 1963.

[18] R. V. Allen, "Automatic earthquake recognition and timing from single traces," *Bull. Seismol. Soc. Amer.*, vol. 68, no. 5, pp. 1521–1532, 1978.

[19] A. Lomax, C. Satriano, and M. Vassallo, "Automatic picker developments and optimization: FilterPicker—A robust, broadband picker for real-time seismic monitoring and earthquake early warning," *Seismol. Res. Lett.*, vol. 83, no. 3, pp. 531–540, May 2012.

[20] M. Baer and U. Kradolfer, "An automatic phase picker for local and teleseismic events," *Bull. Seismol. Soc. Amer.*, vol. 77, no. 4, pp. 1437–1445, 1987.

[21] J. Lee, J.-S. Kim, S. Choi, and Y.-W. Kwon, "A smart device using low-cost sensors to detect earthquakes," in *Proc. IEEE Int. Conf. Big Data Smart Comput. (BigComp)*, Feb. 2019, pp. 1–4.

[22] Z. Li, M.-A. Meier, E. Hauksson, Z. Zhan, and J. Andrews, "Machine learning seismic wave discrimination: Application to earthquake early warning," *Geophys. Res. Lett.*, vol. 45, no. 10, pp. 4773–4779, May 2018.

[23] Z. Li, W. Yang, S. Peng, and F. Liu, "A survey of convolutional neural networks: Analysis, applications, and prospects," 2020, *arXiv:2004.02806*. [Online]. Available: http://arxiv.org/abs/2004.02806

[24] Q. Kong, D. T. Trugman, Z. E. Ross, M. J. Bianco, B. J. Meade, and P. Gerstoft, "Machine learning in seismology: Turning data into insights," *Seismol. Res. Lett.*, vol. 90, no. 1, pp. 3–14, 2018.
[25] T. Perol, M. Gharbi, and M. Denolle, "Convolutional neural network for earthquake detection and location," *Sci. Adv.*, vol. 4, no. 2, Feb. 2018, Art. no. e1700578.
[26] Z. E. Ross, M.-A. Meier, and E. Hauksson, "*P* wave arrival picking and first-motion polarity determination with deep learning," *J. Geophys. Res.*, vol. 123, pp. 5120–5129, Jun. 2018.
[27] S. M. Mousavi, W. Zhu, Y. Sheng, and G. C. Beroza, "CRED: A deep residual network of convolutional and recurrent units for earthquake signal detection," *Sci. Rep.*, vol. 9, no. 1, Dec. 2019, Art. no. 10267.
[28] W. Zhu and G. C. Beroza, "PhaseNet: A deep-neural-network-based seismic arrival-time picking method," *Geophys. J. Int.*, vol. 216, no. 1, pp. 261–273, 2019.
[29] S. M. Mousavi, W. L. Ellsworth, W. Zhu, L. Y. Chuang, and G. C. Beroza, "Earthquake transformer—An attentive deep-learning model for simultaneous earthquake detection and phase picking," *Nature Commun.*, vol. 11, no. 1, pp. 1–12, Dec. 2020.
[30] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent*. Springer, 2015, pp. 234–241.
[31] Y. Ikeda and M. Inoue, "An evacuation route planning for safety route guidance system after natural disaster using multi-objective genetic algorithm," *Procedia Comput. Sci.*, vol. 96, pp. 1323–1331, Sep. 2016.
[32] *National Research Institute for Earth Science and Disaster Prevention*. [Online]. Available: http://www.kyoshin.bosai.go.jp
[33] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A K-means clustering algorithm," *J. Roy. Stat. Soc.*, vol. 28, no. 1, pp. 100–108, 1979.
[34] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2011.
[35] F. Chollet. (2015). *Keras Documentation*. [Online]. Available: https://keras.io
[36] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proc. COMPSTAT*, 2010, pp. 177–186.
[37] S. Haykin, "A comprehensive foundation," *Neural Netw.*, vol. 2, p. 41, Feb. 2004.
[38] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Neural Networks for Perception*. Amsterdam, The Netherlands: Elsevier, 1992, pp. 65–93.
[39] R. Caruana, S. Lawrence, and C. L. Giles, "Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 402–408.
[40] F.-C. Chen, "Back-propagation neural networks for nonlinear self-tuning adaptive control," *IEEE Control Syst. Mag.*, vol. 10, no. 3, pp. 44–48, Apr. 1990.
[41] G. Hinton, N. Srivastava, and K. Swersky, "Neural networks for machine learning lecture 6a overview of mini-batch gradient descent," *Cited*, vol. 14, no. 8, pp. 1–31, 2012.
[42] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980
[43] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

SEONHYEONG KIM received the M.S. degree in computer science from Kyungpook National University, South Korea, in 2018, where she is currently pursuing the Ph.D. degree in computer science. Her research interests include the Internet of Things, mobile computing, and big data. Her current research is about the collection and analysis of big data in various formats.

IRSHAD KHAN received the M.S. degree in computer science from the National University of Computer and Emerging Sciences, Peshawar, 2009. He is currently pursuing the Ph.D. degree in computer science from Kyungpook National Unviserty (KNU), South Korea. His area of expertise includes artificial intelligence, machine learning, and pattern matching. His current research interests include the real-time detection of P and S waves in a seismic waveform.

SEONHWA CHOI received the Ph.D. degree in computer science from Chonnam National University (CNU). Since 2008, she has been working as a Team Manager of disaster information management with the National Disaster Management Research Institute (NDMI). Her research interests include natural language processing and big-data analysis in disaster management areas.

YOUNG-WOO KWON received the Ph.D. degree in computer science from Virginia Tech. He is currently an Associate Professor with the School of Computer Science and Engineering, Kyungpook National University. Prior to coming to KNU, he was an Assistant Professor with the Department of Computer Science, Utah State University. His research interests span distributed systems, mobile computing, and software engineering as applied to earthquake early warning, and disaster big-data analysis.