CrowdQuake+: Data-driven Earthquake Early Warning via IoT and Deep Learning

Aming Wu*, Jangsoo Lee*, Irshad Khan, Young-Woo Kwon ✉
Kyungpook National University, Daegu, South Korea
Email: wuaming@knu.ac.kr, dellhart@knu.ac.kr, irshad.cs@knu.ac.kr, ywkown@knu.ac.kr

2021 IEEE International Conference on Big Data (Big Data) | 978-1-6654-3902-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/BigData52589.2021.9671971

***Abstract*—In recent years, a low-cost micro-electro-mechanical systems (MEMS) acceleration sensor has been widely used for earthquake early warning (EEW). In our previous work, we introduced a networked earthquake detection system, CrowdQuake with three-hundred smartphones' acceleration sensors and a deep-learning based earthquake detection model. For one year's operation, CrowdQuake detected a series of earthquakes and collected various earthquake and non-earthquake data. Based on the successful operation of CrowdQuake, in this paper, we discuss how it can be expanded across the country by addressing the following challenges: (1) sensor deployments for highly dense network, (2) earthquake detection performance using a deep learning model, and (3) high performance and scalable system design for big data processing. The improved system is CrowdQuake+ which can deal with acceleration data sent from 8,000 IoT sensors and detect an earthquake in few seconds using a newly proposed detection model. Moreover, CrowdQuake+ stores all acceleration data sent from sensors and assesses their qualities by calculating noise levels. Then, the collected data are used for deep learning model training, so that its detection performance becomes more accurate.**

***Index Terms*—Earthquake early warning, IoT, Acceleration sensor, Deep learning, Distributed systems**

I. Introduction

With the development and application of the technology of Internet of things (IoT), people have been able to apply cutting-edge technologies to real-time monitoring of many geological disasters, so as to minimize economic losses and casualties [1], [2]. In seismology, however, due to the high requirements of seismic network density and data processing complexity [3], [4], the development of high-performance real-time EEW system is still facing great challenges.

Big data processing technology has become a trend to process large amounts of data continuously recorded on the seismic network in real-time [5]. A pilot seismic network with 300 smartphones was deployed in our previous work, and a seismic detection system based on deep learning called CrowdQuake was used for real-time EEW [6]. The experimental results confirm that the detection approach proposed is feasible, whose performance is significantly better than that of the traditional machine learning (ML) methods[7]. Because of limited coverage of sensors, we need to expand the seismic network density, so as to put them into use nationwide. As the number of data increases, the need for continuous monitoring of all sensors increases, and it becomes more difficult to perform all data processing operations on a single device.

We are considering that IoT sensors are affected by various environmental factors (Traffic, weather, human activities, etc.); these recorded data with the low signal-to-noise ratio (SNR) only produce one or two components of continuous fluctuations, which is likely to be a small earthquake. Nonetheless, the existing ML model cannot accurately detect each component separately and the smiling ground motion, which leads to the missing alarm, interrupting industrial activities or critical transportation systems. Our previous convolutional recurrent neural network (CRNN) model can detect earthquakes in a one-dimensional convolutional network based on 2-second data, but mixed noise data will affect its detection performance, which is one of the main challenges. Therefore, an intelligent detection model is quite essential for the robustness and reliability of EEW.

To solve the above problems, we first adjust the sensor deployment density and adopt the distributed system to process data, which increases the scalability and has the advantages of eliminating the single points of failure and allowing to execute or replace various modules according to needs. Meanwhile, increasing the density of the seismic network is also beneficial to improving the detection accuracy of the EEW system. The CrowdQuake+ can receive and process data from about 8000 IoT sensors. While dealing with different levels of noise data recorded by large-scale sensors, the K-means clustering method is used to classify and analyze the quality of sensors. Also, we use the concept of multi-scale entropy to develop a novel deep learning model called a multi-head convolution neural network (MCNN) to extract the characteristics of multi-scale and high-dimensional data and detect seismic events through 2-second waveforms. Importantly, we utilize different evaluation methods to confirm the performance of the CrowdQuake+ system and MCNN model, respectively. Besides, they present excellent trigger results in practical application. The contributions of the paper are as follows:

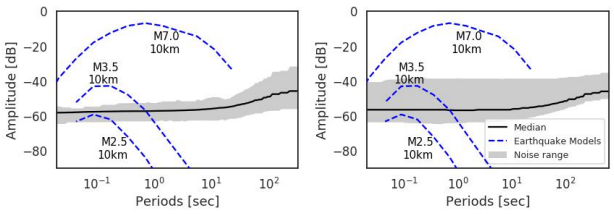
- **Deployments and maintain of the IoT sensors:** We increased the sensor deployment density nationwide by more than 6,000 low-cost MEMS acceleration sensors instead of smartphones running in Korea for more than a year, and intelligently managed these sensors based on K-means clustering algorithms.
- **Distributed system for processing large-scale sensor data:** The data processing system is divided into two

*Both authors contributed equally to this work.
✉ Corresponding author.

main frameworks in accordance with whether buffer pool is required (i.e., batch processing and stream processing) to meet the data requirements of different processes and minimize the delay caused by data processing.

- **Machine learning approach to detect earthquake events:** The detection process is first triggered by STA/LTA to remove redundant data and improve detection efficiency. Besides, the MCNN model is used to detect make decisions.

The rest of this paper is organized as follows. Section II describes the background and motivation for improving the CrowdQuake. Section III shows the details on the extension of CrowdQuake includes the sensors deployment, the adjustment of system structure and the optimization of algorithms. In Section IV, we verified the EEW system and model by various evaluation approaches. In addition, Section V discusses the related work. Finally, the work is concluded in Section VI.



(a) CrowdQuake (b) CrowdQuake+

Fig. 1: Variance of the noise distribution between stages

II. Background and Motivation

An EEW system is an effective means of earthquake prevention and disaster reduction [3], [8]. In our previous work [6], we deployed 300 smartphones equipped with an acceleration sensor and developed a distributed system, CrowdQuake, to detect an earthquake using the ML model. Even though CrowdQuake successfully could detect earthquakes within a few seconds, several challenges were identified to further expand CrowdQuake. In the following discussion, we present lessons learned from CrowdQuake and then address them in the newer system, CrowdQuake+.

A. Sensor Deployment Issues

Based on 300 smartphones deployed in the CrowdQuake, it is not enough to apply the EEW system across the country. As we have a plan to install 8,000 IoT sensors, some problems should be addressed in CrowdQuake+. First, because we used base stations of Korea's wireless communication operator (SK Telecom) to deploy smartphones and IoT sensors, the distribution of base stations is directly proportional to the population density, resulting in unbalanced distribution between suburb and downtown, which can reduce the overall detection performance. Second, the noise characteristics of sensors are affected by various operating environments (e.g., unstable power supply, human activities, and weather). As we install more sensors that are exposed to various operating environments in CrowdQuake+, the range of noise floors

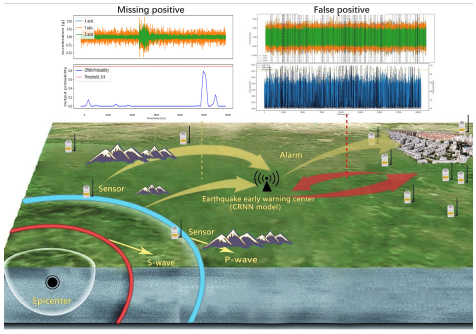


Fig. 2: The false alarm of previous CrowdQuake

becomes wider (Fig. 1), which makes it difficult to detect an earthquake. However, due to the limited resources, searching for appropriate locations to install thousands of sensors is not a proper solution. Therefore, we take a simple strategy in which we deploy IoT sensors as many as possible (up to 8,000) and then use only high-quality sensors for earthquake detection by assessing recorded data every hour.

B. System and Network Issues

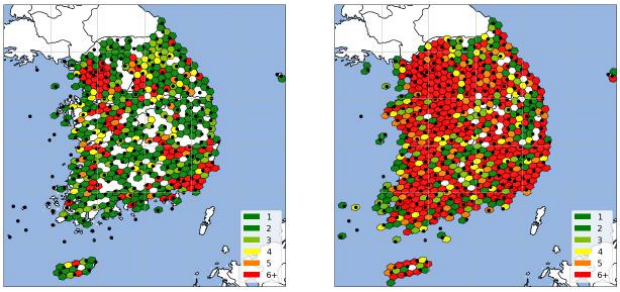
CrowdQuake+ consists of thousands of sensors and multiple server systems for data processing, analysis, and visualization. However, due to the large amount of data transferred from different sensors and heterogeneous operating environments, network delay is unavoidable, it cannot be easily achieved to detect in real-time. Furthermore, to process large amounts of data coming from 8,000 sensors, proper distribution of high-performance computing resources are necessary. However, when distributing processings, because new types of issues (e.g., concurrency-related errors, single point of failure, and processing delay) can be raised, the system should be carefully designed considering the above issues.

C. Detection Model Issues

IoT sensors continuously captures the vibration data (earthquake and noise generated by surroundings and human activities), which makes it a very challenging task to accurately identify an earthquake in less time window. In the CrowdQuake, however, due to the different sensitivity of the horizontal and vertical components of the 3-component data to noise, it is difficult for the CRNN model based on a one-dimensional convolution network to detect small seismic, which will affect the performance of the EEW. Fig. 2 vividly presents the complex noise records of small earthquakes or high noise seismic network areas in low seismic network density areas. Through detection system, false positives are caused. The yellow arrow indicates the small earthquake from the mountain area, but the CRNN may miss the unique characteristics of the seismic, resulting in the loss of positive signals; The red arrow presents that the high noise recorded by the sensor is detected as a seismic event by CRNN.

III. Extension of CrowdQuake

To overcome limitations and improve the performance of the CrowdQuake, we developed a new IoT sensor and re-



(a) CrowdQuake system (b) CrowdQuake+ system

Fig. 3: Sensor distribution map: Different colors represent the distribution density of sensors. Red indicate more than six sensors while dark green refer to only one sensor.

developed the detection system. In Part A, we describe the deployment of sensors. In part B, we present the overall architecture of CrowdQuake+ and the decisions we have taken to improve the detection performance. We discuss the novel detection model and compare it with the previous CRNN model in part C.

A. Community Supported Deployment

The deployment status is shown in Fig. 3, only 300 smartphones were deployed as seismic sensors through the previous CrowdQuake (Fig. 3a). Considering its low coverage and high installation cost, in CrowdQuake+, we have deployed more than 6,000 IoT sensors across the country (Fig. 3b). However, considering some mountainous areas with complex terrain, the coverage density of sensors needs to be improved. Also, it is a complex issue to choose the installation location of about 8000 sensors in Korea only according to the positioning principle of the communication base stations. To overcome this problem, we need to find a national average distribution of facilities as a reference to determine the sensor installation locations. Fortunately, in order to provide all citizens with necessary life services, such as education, security and postal service, the Ministry of Education is distributing these facilities as equally as possible, which provide a scientific basis for the rational deployment of sensors across the country. It enables us to use infrastructures like observation stations, and easy deployment of 8,000 sensors across the country. Our vision is to build a fully deployed earthquake network in the whole country.

B. System Architecture

Fig. 4 shows the overall architecture of CrowdQuake+. In this section, we describe a series of decisions taken to enhance the data transmission, storage, and processing of the system.

1) Sensing Layer: We have set up an IoT device composed of low-cost sensors, which communicate with each other through Cat.M1 in CrowdQuake+. The built-in accelerometer collects data at a rate of 100 Hz (i.e., 100 data points per second) and send them to the networking layer through the communication modem (i.e., Cat.M1), with an average network delay of about 40 ms and a bandwidth of 375 Kbps between the IoT devices. Although the bandwidth of Cat.M1 is not as large as that of a commonly used Long Term Evolution (LTE) and 5G, the performance is satisfactory for a sensor that usually generates about 1kb of data per second. To automatically update the software system running on a device, the new version of the software system is transmitted through over-the-air (OTA). In addition, because the the programs running on the IoT sensors can directly access the hardware system and avoid irrelevant services running in background, which commonly occurs on an Android-based system. Hence, our IoT devices can provide more stable data collection conditions compared to smartphones.

Once a sensor is turned on, it continuously records acceleration data per 10 ms and sends them per second. In the CrowdQuake+, sensors are used only to collecte real-time acceleration data. Those data can be used not only for detecting seismics but also understanding noise patterns surrounding sensors. Moreover, sensor data can be used to monitor the structural health of a building [9]. The collected data is packed into a single message and then sent to the backend system for further processing.

2) Networking Layer: The sensor management is separated from data processing. This networking layer pre-processes sensor data by adding a unique ID and correcting their orders asynchronously through transmission from thousands of sensors, it consists of multiple message queues to reduce the risk due to network or system failure. This separation forms a clear boundary between the management of sensors and the data processing, which also allows the data processing layer to receive data from traditional seismic network operated by the government. Similarly, our sensor data can be forwarded to other systems other than CrowdQuake+.

3) Data Processing Layer: In the previous CrowdQuake, except for data archiving, all processing operations (i.e., sensor connection management, authentication, detection) are completed parallel on the same machine. The parallel operation of this bifurcation process is possible because it only uses 300 smartphones. However, with an increasing number of sensors (up to 8,000) and their heterogeneous operating environments, timely detection has become a significant challenge in CrowdQuake+. Here, acceleration data need to be processed in two ways: stream processing for real-time earthquake detection and batch processing for quality assessment and storing. To that end, we redesign CrowdQuake to meet such requirements.

Messaging in Data Processing Layer: The processes are interconnected through message queues (i.e., Kafka) to form a processing pipeline and generate seismic alarms from the original acceleration records. Whenever a gateway receives a record from the networking layer, a topic is set for the record to enable pub/sub pattern messaging to the acceleration record processing unit. The topic is decided as *network.subnet.usim* (*network* is the type of sensor network, *usim* is the 11 digits assigned by the communication company, *subnet* uses a value of *usim* % 10). Subscription is in the form of topic prefix matching. *subnet* in a topic is intended to use as a unit of parallelism. That is, the records can be divided to be in charge based on *subnet* through processes with the same routine. Finally, all processing results are pushed to the Network

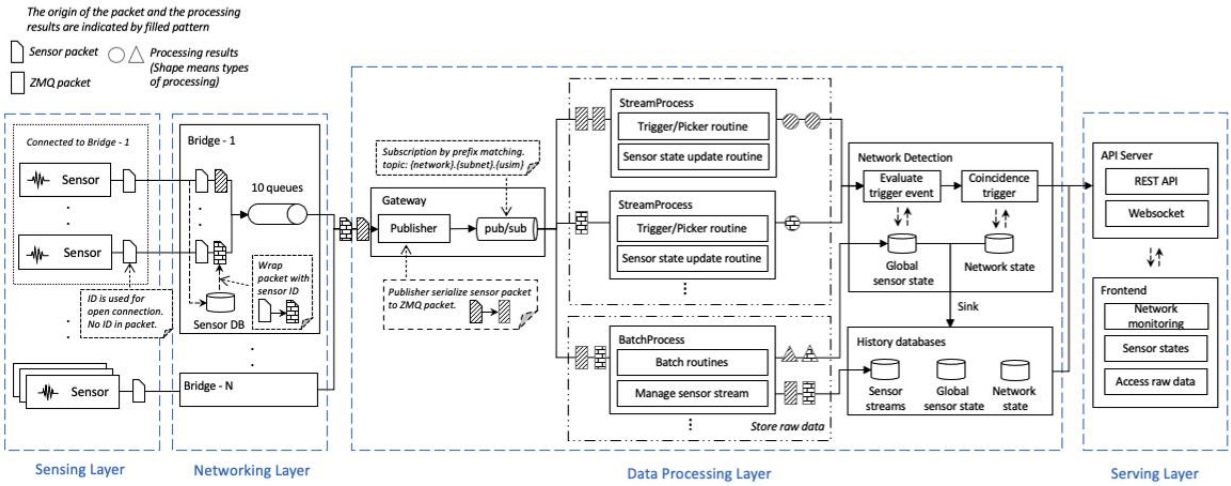


Fig. 4: System overview

Detection Module, sending a decision to the serving layer.

Acceleration Processing: Based on data transmission and processing patterns, we divide routines into two categories according to whether a buffer pool is provided.

`BatchProcess` has a buffer pool and a list of tuple that (*routine*, *length*) where *routine* is operation want to be executed, and *length* is required length to run the *routine*. Each buffer in the buffer pool, named SensorStream, supports timely adjustment when sorting and data omission. When a new ID appears in the process, the batch process assigns a new SensorStream to the sensor, and when enough data collected for a certain routine, BatchProcess execute that routine onto the internal worker process. Current implementation using shared memory for the place buffer pool and executing the routine with one of the processes in the process pool.

`StreamProcess` manages triggers to be run and the state of each sensor it handles. The state is the storage of a variable that is used to run the trigger. For example, the ML model saves a one-second record just right before the current one to make two seconds length of the input. If data is omitted or out-of-order, the sensor status will be reset and restarted. This decision has been made based on the principle that data freshness is more important than data consistency.

Event Processing: There are two types of events from previous stage, including data qualities and trigger events. We first filter trigger events based on the sensor data quality. If the quality of sensor data is too bad to detect earthquakes, trigger events caused by low-quality data are ignored. We finally make an earthquake event by associating triggers.

4) Service Layer: In this layer, CrowdQuake+ provides an API server and a web front end to enable real-time monitoring and receive alerts from the data processing layer. Also, statistics of the system and detected events are provided. To that end, CrowdQuake+ provides the following API and views: (i) real-time notification of an earthquake, (ii) monitoring of CrowdQuake' network and system status, and (iii) historical data including raw acceleration data and event data.

Management and Maintenance of CrowdQuake+: Because data qualities are affected by the installation environment of sensors (i.e., human activities, and mechanical problems), we need to monitor the status and noise level of each sensor carefully. However, due to the wide range of sensor deployment, it is impossible to manually manage all sensors. Therefore, we adopt the K-means clustering method to detect abnormal sensors and replace them in time. Also, noisy data recorded by such abnormal sensors are excluded in the detection model to improve the the overall performance and accuracy of CrowdQuake+.

`The Idea of K-means Clustering`: A given data set is divided into k clusters according to the distance among samples. Then make the points in the cluster as close as possible and the distance among clusters as large as possible [10]. We assume that the cluster is divided into $C = \{C_1, C_2, ..., C_k\}$, the goal is to minimize the square error E:

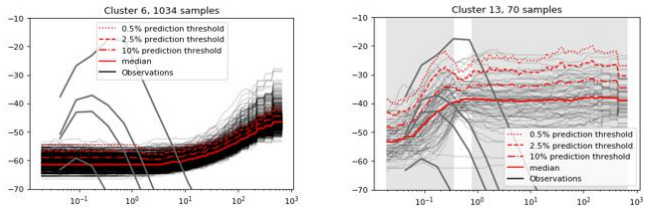
$$E = \sum_{t=1}^{k} \sum_{x \in C_t} \|x - \mu_t\|_2^2 \tag{1}$$

Where μ_t is the mean vector of Cluster C_t:

$$\mu_t = \frac{1}{|C_t|} \sum_{x \in C_t} x \tag{2}$$

Since it is an NP-hard problem, it is hard to find the minimum value, and thus we use a heuristic iterative method.

`Experimental Process`: To cluster different sensors, power spectral density (PSD) and probability density function (PDF) [11] are usually used as the quality criterion baseline. We train the K-means clustering model using recorded data, and the PSD is calculated based on these datasets, from which We extract features. As a result, we generate 6,978 training datasets and 12 of features. Then we scale them according to the variance of the data features and normalize them by removing the mean value. Since the key of the K-means clustering algorithm is choosing proper k, we use scikit-learn [12] to select an optimal cluster $k = 16$ through continuous iterative optimization.



(a) Cluster 6: Low noise (b) Cluster 13: High noise

Fig. 5: Analysis based on K-means clustering

`Clustering Results`: Fig. 5 displays two samples of the clustering result. For example, the sensors belonging to Cluster 6 (Fig. 5a) have low and constant background noise, while Cluster 13 (Fig. 5b) has a high noise level and an unstable status. Then, we manually regroup sixteen clusters into three groups, such as GREEN (excellent), YELLOW (good), and RED (poor), and then only GREEN and YELLOW sensors are used for earthquake detection.

C. Earthquake Detection

In this part, we propose a new deep learning algorithm based on each component's independent detection to improve the detection accuracy for small earthquake events and better solve the problem of false alarms caused by blind noise triggering.

1) Earthquake Detection Process: Considering the increase of complex data recorded by IoT sensors, in CrowdQuake+, we proposed a multi-task seismic detection algorithm for decision-making: We first adopt STA/LTA algorithm, based on the vector sum of 3-component data, and set a reasonable threshold to preliminarily filter effective seismic data. However, STA/LTA algorithm will receive different levels of noise interference and generate false triggers. The recognition rate of microseismic events needs to be improved. Consequently, based on the MCNN model, we further identify whether the trigger event is a seismic and transmit the result to the server.

2) Detection Model Based on the CRNN: The previous CrowdQuake system identify earthquake events using CRNN according to the 2-second data. The CRNN combines the advantages of CNN and RNN without human interventions (such as the ANN model) while capturing the implicit features of the acceleration data. Also, the CRNN model completes the detection task on the one-dimensional convolutional network based on the 3-component data (X, Y and Z) of 2 seconds, which split into two sub-samples of size $\frac{n}{2} \times 3$ (i.e., n is the total number of samples in a 2-second window, and 3 is the three components) where the output of the first sample is feed as an input with the second sample as the final output.

3) Design of the MCNN Model: The seismic time series have different features in different components, a traditional one-dimensional convolution network is often with a single scale, and the characteristics of multivariate time series are not fully exploited, which is easy for the EEW to cause a false alarm and the omission of small ground motions. Simultaneously, because there is a difference in the sensitivity of the 3-component seismic sequence to noise, it is necessary to develop a novel detection model through which the seismic from human activities can be accurately distinguished. The MCNN is designed to detect the multiscale and high-dimensional seismic sequences based on the concept of multiscale entropy (MSE). The network structure is shown in Fig. 6.

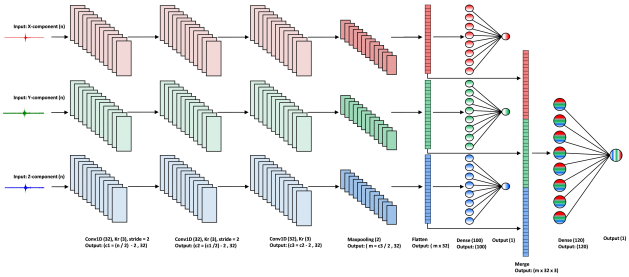


Fig. 6: MCNN architecture

The first three layers are 1D convolutional layers containing 32 filters with a kernel size of three (i.e., 1×3) and a stride of two in the first two layers while the one in the third layer is without padding. To keep the low complexity of the model, we keep the former configuration layers. The forward propagation of the one-dimensional CNN for a component (i.e., X-component) is as follows:

$$x_i^l = b_i^l + \sum_{j=1}^{N_l-1} \mathrm{Conv}\,1D\left(w_{ji}^{l-1}, o_j^{l-1}\right) \tag{3}$$

Here x_i^l is the i^{th} neuron at layer l , b_i^l is the bias of the i^{th} neuron at l layer, o_j^{l-1} is the input to the current layer (i.e. output of the j^{th} neuron at layer $l-1$), w_{ji}^{l-1} is kernel from the j^{th} neuron at layer $l-1$ to the i^{th} neuron. We used the ReLU activation function for non-linearity in convolutional layer.

$$y_i^l = f\left(x_i^l\right) \tag{4}$$

Where $f(z) = max(0, z)$.

Max pooling is the fourth layer with a stride of two to extract the prominent features and reduce the dimension. The output (e.g., 22×32 for a 2-second input) of max-pooling is then flattened ($F_{i\in(1,2,3)}$) and input to the fully connected dense layer with ReLU activation function. We produced the probability $P \in (0, 1)$ of each input separately using the sigmoid activation function's output layer for each component. For the combined final output, we merge all the three flatten layers as the input nodes of the final output and feed them to another fully connected dense layer for the final output.

$$M = \mathrm{Merge}\left(F_1, F_2, F_3\right) \tag{5}$$

Here the size of M for a 2-second input is $(22 \times 32 \times 3)$.

We implement the model in Keras, which is a Python deep learning library. As our model is a binary classifier, we use the binary cross-entropy loss function to minimize parameters of the training model. Furthermore, we use Adam optimizer and mini-batch processing for weight and parameter optimization [13]. To avoid overfitting, we utilize a dropout layer with the probability of 0.5 [14].

IV. Experience and Analysis

The behavior of an EEW system depends on an easy maintenance data processing system, a high-density seismic network and high-precision detection algorithm. This section is organized as follows: In the part A, we implemented the networking layer of CrowdQuake+ using different messaging queues to evaluate the system. Through the experimental verification of different data sets, we examine the detection accuracy of the MCNN model and further compare its effectiveness with that of CRNN models in the part B.

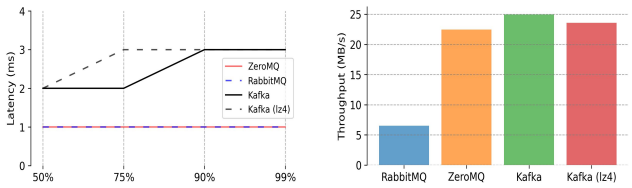


Fig. 7: The latency and throughput of four messaging queues

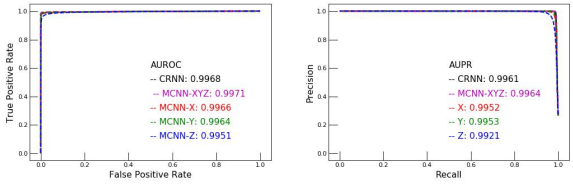
A. System Evalution

To evaluate our system, we implemented the networking layer of CrowdQuake+ using these messaging queues including RabbitMQ, ZeroMQ, Kafka, Kafka with compression. We measured the latency and throughput of them, and Fig. 7 shows the evaluation results. In latency comparison, Kafka's latency is longer than ZeroMQ and RabbitMQ while in throughput comparison Kafka's throughput is outperforming RabbitMQ and ZeroMQ. When compressing data in Kafka, it incurs longer latency and less throughput. Based on our evaluation, the current version of CrowdQuake+ uses Kafka because of its powerful failure handling mechanisms.

B. Model Evaluation

1) Data Collection and Preprocessing: Based on different data sets (including earthquake and non-earthquake data sets), we compare and evaluate our MCNN model with the CRNN model. The earthquake dataset contains high-quality data, synthesis data (where a noise template is mixed) downloaded from the National Research Institute of Earth Science and Disaster Prevention (NIED) [15]. These data are all earthquake events with the PGA no less than 0.05g. The purpose is to further evaluate the model's performance in the virtual environment. We also selected four lastest earthquakes recorded by IoT sensors; Non-earthquake data includes hours of human activity and blind noise collected from IoT sensors. Except preprocessing all the data sets with a sampling rate of 100Hz and converting the unit into g, we use a 0.3Hz-30Hz band-pass filter to filter the data.

2) Evaluation Methods: **Method 1:** In CrowdQuake+, the core idea of MCNN model is a binary classification. Consequently, in this experiment, we evaluate the performance of the model through a confusion matrix. In order to avoid the interferences caused by the imbalance between positive and negative samples in the actual test samples (positive means earthquake and negative means non-earthquake), we further



(a) AUROC (b) PR

Fig. 8: Models' AUROC and PR curves (Test1)

evaluate the performance through Area Under the ROC (AUROC) and Precision-recall (PR) curves, which are particular indicators for evaluating the problem of binary classification.

Method 2: We also tested the actual triggering performance of the MCNN, simulated the seismic by our sensors, observed the number and reliability of sensors triggered by the two models in practice.

3) Model Results Analysis: **Test1:** We trained 100 epoch models on 70% of the training set and tested them on the remaining 30% of the data sets. We compared the test results of the MCNN proposed with the previous CRNN (As shown in Table I). AUROC and PR curves of the two models are shown in Fig. 8a and Fig. 8b. The performance of the new MCNN model for each acceleration component exceeds 98.5%. Although the detection results based on the single component are not enough to make a decision for each sample, they can avoid false positives triggered by the high noise of a single component. In addition, the overall detection performance of the MCNN (i.e. MCNN-XYZ) model for the 3-component data is slightly better than the CRNN model.

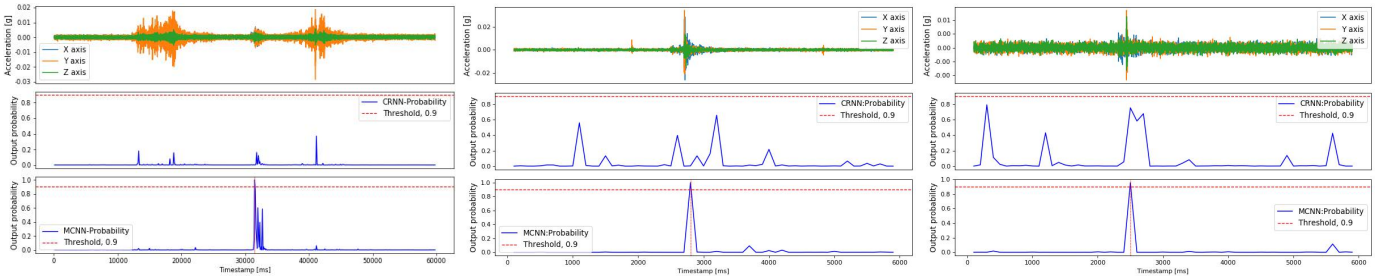
TABLE I: Models performance on 30% data (Test1)

Model	TP	FP	TN	FN	Accu:	Pre:	Rec:
CRNN	71298	345	200907	962	99.52	99.51	98.67
MCNN	71446	261	200991	814	99.61	99.64	98.87
MCNN-X	70902	1261	199991	1358	99.04	98.25	98.12
MCNN-Y	71121	1839	199413	1139	98.91	97.48	98.42
MCNN-Z	69767	1460	199792	2493	98.55	97.95	96.55

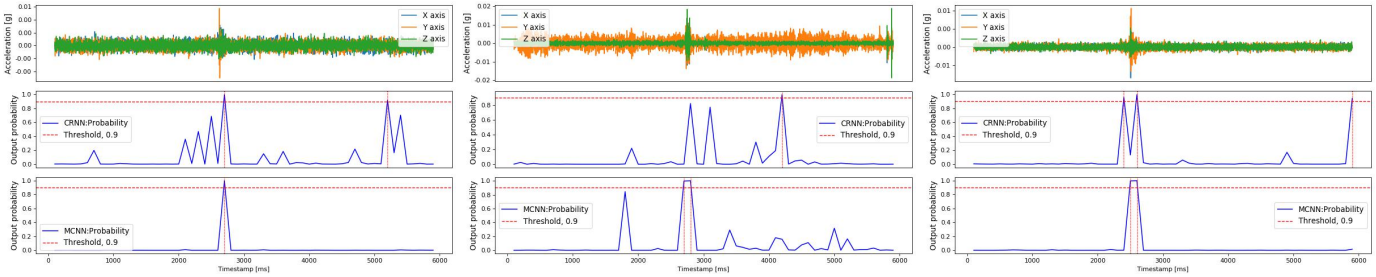
TABLE II: Models performance on 30% data (Test2)

Model	TP	FP	TN	FN	Accu:	Pre:	Rec:
CRNN	71398	2658	75906	391	97.97	96.41	99.46
MCNN	71483	1296	77268	306	98.93	98.22	99.57
MCNN-X	71051	1504	77060	738	98.51	97.92	98.97
MCNN-Y	71045	1179	77385	744	98.72	98.37	98.96
MCNN-Z	70979	3540	75024	810	97.11	95.25	98.87

Test2: In the second experiment, we trained both models on high-quality earthquake and synthesized data while including 30% of the low-quality earthquake data and some portions of non-earthquake data from our sensors for the test to keep a balance in both the classes. We observed more performance degradation in the CRNN compared with MCNN (Table II). Because of the variable characteristics of the sensor, the length of the input window and the lightweight structure of the model, the seismic detection task is challenging. Therefore, there is always a trade-off between the recall and accuracy of the model in question, especially because CRNN has a lower recall rate and a higher accuracy. On the other hand, when we use

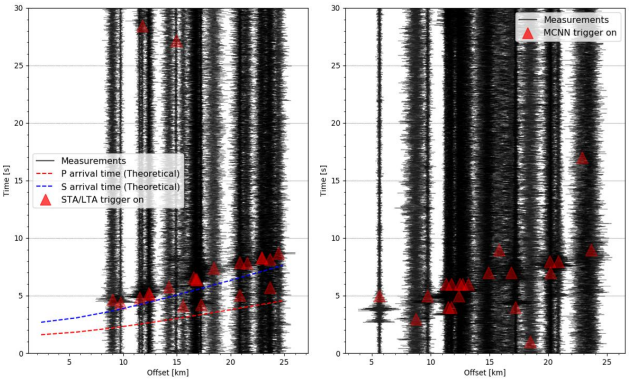


(a) Sensor-1753_20200427 (27/04/2020) (b) Sensor-01232971368(28/08/2021) (c) Sensor-01232973215(29/08/2021)



(d) Sensor-01232973026(28/08/2021) (e) Sensor-01227587566(29/08/2021) (f) Sensor-01232971546(29/08/2021)

Fig. 9: Each sub-figure from top to bottom: 3-component acceleration (X,Y, and Z); CRNN probabilities; MCNN probabilities; (a)-(f) represent detection results by the CRNN and MCNN models based on the latest earthquake events in South Korea



(a) STA/LTA triggers (b) MCNN triggers

Fig. 10: The CrowdQuake+ trigger results are from the STA/LTA and MCNN model based on the latest earthquake

medium-level non-earthquake data for training, the recall rate is high, and the accuracy rate is low. The behaviour of data-driven models is changed based on the underlying dataset. However, through the test, we observed that the behavioural change of CRNN was more significant than that of MCNN.

To further verify the performance of the MCNN model, we collected 6-second ground motion data and tested several latest seismic in Korea, the result examples are shown in Fig. 9. By comparing Fig. 9a-9c, it is obvious that MCNN model can accurately identify earthquakes in a short time window without missing alarms, while the CRNN can hardly detect these earthquakes with small PGA values; similarly, Fig. 9d-9f clearly show that the CRNN model is prone to trigger caused by noise in micro-earthquakes. These results indicate that the MCNN model has excellent detection ability and robustness to micro-ground motion events. Given the latest M2.2 earthquake event in Gyeongsang, South Korea, we use STA/LTA and MCNN to test the earthquake from X, Y, Z components, respectively. Fig. 10 presents the trigger results, the number of trigger sensors detected by the MCNN model is significantly more than STA/LTA within 15 kilometers from the epicenter, which reveals that our model has more reliable in the practical system application.

V. Related Works

In the past decades, with the development of seismological theories and signal processing technology, different countries and institutions began realizing the EEW through MEMS sensors and smartphones [16], [17], [18], [19]. However, the deployment scale of these sensors and the network structure of the system are not considered. Sensor deployment and data processing capability are the key to the implementation of the EEW. There are MEMS sensors installed in the NetQuake system developed by the United States Geological Survey (USGS), but it is mainly concentrated in California [20]. Low-cost MEMS sensors are mainly used in Quake-Catcher Network (QCN) and Community Seismic Network (CSN), which need to be embedded in computers or installed in buildings to detect seismic events [21], [22]. Myshake is an EEW system based on smartphones. Because the device is easy to carry and operate, it has unique advantages in sensor deployment and network communication. An accelerometer is applied to the EEW system, which is straightforward to popularize in densely-populated cities and suburbs. However, the cost of a smartphone seismograph is relatively high, and its sensitivity needs to be improved for below M5 [23].

With the expansion of the sensor deployment scale, seismic data processing as well as analysis technologies are facing new challenges. The ML provides many new methods to

extract data features, which greatly saves the cost of online data processing for the EEW systems [24]. The PhaseNet is a deep learning model used to identify P-waves, S-waves, and noise signals in a 30-second waveform [25]. Similarly, the EQTransmers is another seismic detection model with a longer input window based on the deep learning [6]. Both detection models mentioned above detect the seismic waveform through a long time window, which lacks data processing speed. The CrowdQuake system proposed by us takes into account the impact of high-noise data recorded by sensors on the performance of the detection model, which tries to improve the data transmission and processing speed meanwhile avoiding false triggers and delays and has been deployed nationwide.

VI. Conclusion

This paper expanded our previous EEW system (CrowdQuake) using 300 smartphones to support 8,000 IoT sensors. Through one year of operation of CrowdQuake+, we have identified several technical challenges to process tremendous amounts of acceleration data in real-time and then improved our system, which is CrowdQuake+. Compared to CrowdQuake, more sensors have been installed at many different places exposed to various types of noise sources that can decrease the overall detection performance of CrowdQuake+. To that end, we designed our system in a distributed way to concurrently process sensor data and employed a quality assessment technique to exclude low quality sensor data in an earthquake detection algorithm. Moreover, to improve the detection performance, we newly designed the deep learning-based detection model, MCNN, that takes individual accelerometer components as input. Our experimental result shows that CrowdQuake+ can detect earthquakes greater than M2.5 in four to five seconds, which is considerably shorter than a traditional EEW system. In future research, we will further expand CrowdQuake+ to support more sensors and explore how CrowdQuake+ can be integrated with a traditional EEW system.

Acknowledgment

This research was supported by Basic Science Research Program through the National Research Foundation of Korea funded by the Ministry of Education (NRF-2021R1I1A3043889) and by the Development of Earthquake, Tsunami, Volcano Monitoring and Prediction Technology (KMA-135002988).

References

[1] I. Lee and K. Lee, "The Internet of Things (IoT): Applications, investments, and challenges for enterprises," *Business Horizons*, vol. 58, no. 4, pp. 431–440, 2015.

[2] J. A. Strauss and R. M. Allen, "Benefits and costs of earthquake early warning," *Seismological Research Letters*, vol. 87, no. 3, pp. 765–772, 2016.

[3] T. H. Heaton, "A model for a seismic computerized alert network," *Science*, vol. 228, no. 4702, pp. 987–990, 1985.

[4] C. Satriano, Y.-M. Wu, A. Zollo, and H. Kanamori, "Earthquake early warning: Concepts, methods and physical grounds," *Soil Dynamics and Earthquake Engineering*, vol. 31, no. 2, pp. 106–118, 2011.

[5] S. Colombelli, R. M. Allen, and A. Zollo, "Application of real-time GPS to earthquake early warning in subduction and strike-slip environments," *Journal of Geophysical Research: Solid Earth*, vol. 118, no. 7, pp. 3448–3461, 2013.

[6] 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 *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020.

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

[8] H. Zhang, X. Jin, J. Li, Y. Wei, and H. Zhu, "Progress of research and application of earthquake early warning systems (EEWS)," *Prog Geophys*, vol. 28, no. 2, pp. 706–719, 2013.

[9] Q. Kong, R. Allen, M. Kohler, T. Heaton, and J. Bunn, "Structural health monitoring of buildings using smartphone sensors," *Seismological Research Letters*, vol. 89, 01 2018.

[10] K. Krishna and M. N. Murty, "Genetic K-means algorithm," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 29, no. 3, pp. 433–439, 1999.

[11] D. E. McNamara and R. P. Buland, "Ambient noise levels in the continental United States," *Bulletin of the seismological society of America*, vol. 94, no. 4, pp. 1517–1527, 2004.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in Python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[13] F.-C. Chen, "Back-propagation neural networks for nonlinear self-tuning adaptive control," *IEEE control systems Magazine*, vol. 10, no. 3, pp. 44–48, 1990.

[14] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[15] "National Research Institute for Earth Science and Disaster Prevention," "http://www.kyoshin.bosai.go.jp".

[16] P. Pierleoni, S. Marzorati, C. Ladina, S. Raggiunto, A. Belli, L. Palma, M. Cattaneo, and S. Valenti, "Performance evaluation of a low-cost sensing unit for seismic applications: field testing during seismic events of 2016-2017 in Central Italy," *IEEE Sensors Journal*, vol. 18, no. 16, pp. 6644–6659, 2018.

[17] A. D'Alessandro, R. D'Anna, L. Greco, G. Passafiume, S. Scudero, S. Speciale, and G. Vitale, "Monitoring Earthquake through MEMS Sensors (MEMS project) in the town of Acireale (Italy)," in *2018 IEEE International Symposium on Inertial Sensors and Systems*, 2018.

[18] Y.-M. Wu, H. Mittal, T.-C. Huang, B. M. Yang, J.-C. Jan, and S. K. Chen, "Performance of a low-cost earthquake early warning system (P-alert) and shake map production during the 2018 Mw 6.4 Hualien, Taiwan, earthquake," *Seismological Research Letters*, vol. 90, no. 1, pp. 19–29, 2019.

[19] J. Fu, Z. Li, H. Meng, J. Wang, and X. Shan, "Performance evaluation of low-cost seismic sensors for dense earthquake early warning: 2018–2019 field testing in southwest China," *Sensors*, vol. 19, no. 9, p. 1999, 2019.

[20] J. Luetgert, D. Oppenheimer, and J. Hamilton, "The netquakes project-research-quality seismic data transmitted via the internet from citizen-hosted instruments," in *AGU Fall Meeting Abstracts*, vol. 2010, 2010, pp. S51E–03.

[21] E. S. Cochran, J. F. Lawrence, C. Christensen, and R. S. Jakka, "The quake-catcher network: Citizen science expanding seismic horizons," *Seismological Research Letters*, vol. 80, no. 1, pp. 26–30, 2009.

[22] R. Clayton, T. Heaton, M. Chandy, A. Krause, M. Kohler, J. Bunn, R. Guy, M. Olson, M. Faulkner, M. Cheng *et al.*, "Community seismic network," *Annals of Geophysics*, vol. 54, no. 6, pp. 738–747, 2011.

[23] Q. Kong, Q. Lv, and R. M. Allen, "Earthquake early warning and beyond: Systems challenges in smartphone-based seismic network," in *Proceedings of the 20th International Workshop on Mobile Computing Systems and Applications*, 2019, pp. 57–62.

[24] T. Perol, M. Gharbi, and M. Denolle, "Convolutional neural network for earthquake detection and location," *Science Advances*, vol. 4, no. 2, p. e1700578, 2018.

[25] W. Zhu and G. C. Beroza, "PhaseNet: a deep-neural-network-based seismic arrival-time picking method," *Geophysical Journal International*, vol. 216, no. 1, pp. 261–273, 2019.