

Supplementary Materials for

MyShake: A smartphone seismic network for earthquake early warning and beyond

Qingkai Kong, Richard M. Allen, Louis Schreier, Young-Woo Kwon

Published 12 February 2016, *Sci. Adv.* **2**, e1501055 (2016)

DOI: 10.1126/sciadv.1501055

This PDF file includes:

Data collection: The MyShake application
Classifier analysis: Detecting earthquakes on a phone
Network detection algorithm
Estimate warning time for Katmandu, Nepal
Fig. S1. MyShake activity 1 November 2014 to 28 February 2015.
Fig. S2. Example earthquake record used to train the ANN classifier algorithm.
Fig. S3. Structure of ANN classifier algorithm.
Fig. S4. Receiver operating characteristic curve.
Fig. S5. Phone trigger times versus epicentral distance.
Table S1. Accuracy score for ANN classifier with 10-fold cross-validation.
Table S2. Simulated network detection performance for U.S. earthquakes.
Table S3. Simulated network performance for various phone densities.
Legends for movies S1 to S4
References (34–39)

Other Supplementary Material for this manuscript includes the following:

(available at advances.sciencemag.org/cgi/content/full/2/2/e1501055/DC1)

Movie S1 (.mp4 format). Network detection animation for the 2014 M5.1 La Habra earthquake.
Movie S2 (.mp4 format). Network detection animation for the 2004 M6.0 Parkfield earthquake.
Movie S3 (.mp4 format). Simulation of network detection with no earthquake.
Movie S4 (.mp4 format). Simulation of network detection for an M6 earthquake.

Supplementary Materials:

- MyShake: A smartphone seismic network for earthquake early warning and beyond
- MyShake: Smartphone based earthquake alerts

Qingkai Kong, Richard M. Allen, Louis Schreier, Young-Woo Kwon

1 Data collection: The MyShake application

"MyShake" is an android application used to release to private/personal phones. It was released to 75 phones in November 2014 (Fig. S1). This test release was aimed at student volunteers on the UC Berkeley campus. The trigger algorithm at the time consisted of a simple STA/LTA algorithm (34). The application first requires the phone to remain stationary for 30 minutes, meaning the acceleration is minimal and most likely the phone is sitting on a stationary surface. When it meets this requirement, the phone enters into "steady state". The ratio of short-term average (STA) and long-term average (LTA) on any of the 3-components must then exceed a threshold to trigger. When it does, trigger information was immediately sent to CPC including the phone location, time of the trigger, phone ID, and the maximum amplitude. A total of 5 minutes of data was also stored locally on the phone from 1 minute before the trigger to 4 minutes after. A ring buffer stores the last minute of accelerometer data in memory at all times for this purpose. The application also uploads state-of-health (SOH) information every 2 hours and can receive updates and triggers from the CPC. The SOH information provides us with basic information about the number of phones running the application, their location, lifetime of the app, etc. We can also update/change the settings of the application on an individual phone or all phones from the CPC, for example changing the trigger parameters. Finally, we can trigger recording on a phone from the CPC. Either individual phones or the entire network can be triggered to record for a period of time. The waveform data was only uploaded when the phone was plugged into power and had a Wi-Fi connection to minimize power and data-plan usage. All these parameters can be modified remotely. We collected four months of triggered human activity data for our training and testing dataset (Fig. S1). During this period 17600 triggers (all due to human activities) were uploaded to our CPC.

Accurate time is key for all data. The drift in the internal clock on the phones is unacceptable for earthquake-related applications, typically ranging from 0.4 – 8.6 sec/day (35). Thus, geographically distributed nodes need to synchronize their clocks. In the last decade, much research has been conducted to synchronize different internal clocks by referring external signal sources such as power lines, FM radio, Wi-Fi, mobile station, etc. Of them, Network Time Protocol (NTP) is the most commonly used clock synchronization protocol. With a very low network and computation cost, NTP is able to synchronize all the participating nodes within a few milliseconds. In the MyShake application all the accelerometer data is associated with its local device clock, so we synchronize them to Coordinated Universal Time (UTC) via NTP. The MyShake application synchronizes its local clock every 1 hour, thereby minimizing network and computation cost while ensuring sufficient clock accuracy at all times.

Power usage of the application is also important. Careful selection of which sensors to use and when is needed to reduce the power needs to a level that would not impact normal daily smartphone use. Our goal was an application that could continuously run in the background and

still only require the phone to be charged once per day for most/typical phone users. Working within these power requirements, we found that it is possible to monitor the accelerometer data continuously all day. However, it is not possible to continuously use the GPS unit, as it is very power hungry. Instead, we only access GPS at specific times when needed. For the initial 2014 release we only attempted to obtain a location when the phones triggers. When a location request is made to the phone, it returns the best available location. If a GPS location is available it is returned. If not, then the location based on triangulation with cell phone towers is used, if not, then the last available location is used.

The current version of MyShake that we plan to release publically is modified to add the classifier analysis developed to distinguish earthquake from non-earthquake motions, and the use of GPS location has been modified. We continue to have the same initial STA/LTA trigger requirement, after the STA/LTA triggered, we use 2-sec data windows with a 1-sec step to calculate the three key features (IQR, ZC and CAV) up to 10 sec after the STA/LTA trigger. The calculated features in each time window are fed into the Artificial Neural Network (ANN) detector (on the phone) to determine if it is a likely earthquake or not. This two-step approach is implemented so that we do not increase the power requirements, since the STA/LTA method is a simple and low cost computation method. The approach to determining location has also been improved by determining the best available location at the time the phone enters steady state. Now, when the phone enters steady state, the application will try to sample the GPS location. It may take a few seconds to minutes before it gets a stable GPS location. Since phones typically sit in steady state for some time (while sitting on a desk or charging over night) it is unlikely that a trigger occurs in the first few seconds or minutes. If for some reason the phone cannot get the GPS location, e.g. the phone is inside a large building, then the cell phone network location that based on cell phone towers is used. The phone then stores the best available location for the duration of the steady state phase and associated it with the other trigger information when the phone next moves.

2 Classifier analysis: Detecting earthquakes on a phone

We used three types of data for training, validating and testing our classifier. Firstly, normal human activity data collected from the MyShake November 2014 release for four-month period shown in Fig S1. For waveforms to be uploaded, the phone must be stationary, and then move to trigger the STA/LTA algorithm as described above. 10 seconds of data immediately following the human trigger is used in our analysis. We used the first three months of data to train and validate the algorithm, and the last month was kept for final testing.

The second type of data consists of earthquakes recorded on smartphones that were placed on a shake table. These include 241 3-component records from 45 shake table tests runs. The input waveforms into the shake table were past earthquakes with amplitudes rescaled to satisfy the displacement capabilities of the shake table. We only selected the strongest portion of the waveforms recorded by the smartphones, see an example in Figure S2. We focus on the strongest portion of the waveforms, as it is difficult for our classifier to distinguish weak earthquake shaking from human activities. This dataset was used entirely for the training and validation phase.

The third type of data also consists of earthquakes, but recorded on regional seismic networks in Japan and US. It was first modified to replicate waveforms recorded on a

smartphone. To do this we first converted the 24-bit data to 16-bit data, then we added a smartphone noise record from the noise floor tests to produce accelerometer records similar to what we would record on a phone laying on a sturdy table during the event. Phones are not expected to trigger on the initial low-energy P-waves, especially for smaller earthquakes, instead to trigger on the larger amplitude portions earthquake shaking. We therefore selected windows of data from only the strongest portion of shaking (e.g. Fig. S2). We used strong motion data from Japan's KiK-Net and K-Net to train and validate our algorithm. Data with horizontal peak amplitude greater than 0.2g for the period from January 1, 1996 to February 1, 2015 was downloaded from NIED (National Research Institute for Earth Science and Disaster Prevention). A total of 317 3-component records from 203 events were selected. To further test the performance of the algorithm, we used earthquake data from the California Integrated Seismic Network (CISN.org). We used 389 3-component records within 30 km of the earthquake epicenter from 5 events that obtained from CESMD (Center for Engineering Strong Motion Data), NCEDC (Northern California Earthquake Data Center), and SCEDC (Southern California Earthquake Data Center), the results is showing in Table 1.

All data was first high-pass filtered in a simulated real-time manner using the method described in (36). A range of characteristics in overlapping 2-sec data windows was calculated with 1-sec step. We experimented with using different window lengths and steps and found this to be the best compromise between having more data and keeping the window short to detect earthquakes more rapidly. 18 different features including frequency features, amplitude features, and statistical features was tested. All features had low to moderate computational requirements making it feasible to rapidly determine their values on a phone.

Since there were far more data points from human activities than that from earthquake data, this imbalance of classes could affect our classifier (37). In order to create a dataset with equal classes, we used the kmeans cluster method (37) to group the human activities into a number of clusters, with the number of clusters being equal to the number of the earthquake data points. The centroid of the cluster was taken to represent human activity data. This not only created a balanced dataset for us to train our classifier, but also reduced the computation burden during the training.

We selected the best 3 features to distinguish between earthquake and non-earthquake data using greedy forward feature (37) selection. They are the interquartile range (IQR) between the 25th and 75th percentile of the acceleration vector sum, the zero crossing rate from the component with the highest value (ZC), and the cumulative absolute velocity (CAV) of acceleration vector sum. IQR is an amplitude parameter that shows the middle 50% range of amplitude of the movement. ZC is a simple frequency measure. CAV is a cumulative measure of amplitude on the three components in the time window and is determined as follows:

$$CAV = \int_0^2 |a(t)| dt$$

Where $a(t)$ is the vector sum of the 3 components acceleration.

An ANN (artificial neural network) approach is used to classify a particular data window as an earthquake or not an earthquake. Each feature was first scaled to a range of 0 to 1. We used an ANN with one hidden layer and completed a grid search to test different numbers of neurons. Best performance is achieved when the ANN has 1 hidden layer with 5 neurons (Figure S3) with a standard sigmoid activation function (37) defined as:

$$s(x) = \frac{1}{1 + e^{-x}}$$

The ANN was trained and validated using 3 months of human activity data, and earthquake data from shake table tests and Japanese events. The dataset was split multiple times using 70% of the data for training and 30% for testing for cross-validation tests. The accuracy of the classifier when applied to the test datasets is very good, showing 98% to 99% accuracy each time (Tables S1, and Fig S4).

Our trained ANN classifier algorithm was tested by applying it to a dataset consisting of data that was not used in the training/validation process. This contained the last month of MyShake human activity data (February 1 to 28, 2015), and data from large US earthquakes modified to represent waveforms recorded on smartphones. Note that no selection criteria were applied to the US earthquake data (recall that for the Japan earthquake data, we only selected the stations have clear large amplitudes). We applied the classifier to all available waveforms, and the results of this validation are shown in Table 1 and described in the main text.

3 Network detection algorithm

Our first-generation network detector identifies multiple triggers in a space-time cluster, and is based on the approach used in our ElarmS-2 earthquake early warning algorithm for traditional regional seismic networks (22). We stored triggers for 20 seconds and look for 4 or more triggers within a 10 km radius region that can be associated. We require greater than 60% of operating phones to have been triggered within 10 km of the location of the event for an event to be declared (the estimated event location is the centroid of the locations of the triggered phones.). The origin time is assumed to be that of the first phone to trigger. The magnitude is estimated based on the peak ground acceleration of the triggered phones as described in the main text. Triggers from phones at greater than 10 km must then fall within a defined space-time region to be associated with the event (Fig. S5).

We used simulated phone triggers from two earthquakes to test the performance of the algorithm: 2014 La Habra M5 earthquake and 2004 M6 Parkfield earthquake. See the Fig. 6, Movies S1, and S2 for the detection of these two earthquakes, and Table S2 for the performance of the algorithm. In these simulations we assume zero latency due to processing and network transmission. We estimated the actual latency that will be introduced into the system due to the processing on the phone and network transmission. First, to estimate the processing delay of the ANN on the phone we did a test run for one night and found the average processing time is to be 4.5 milliseconds. Second, the transmission of the trigger data from phone to CPC is via UDP (User Datagram Protocol), which is a common choice for time-sensitive applications. We found that the average delay time of transmitting the data from the phone to the CPC via UDP is 50 milliseconds.

In addition to the simulated phone triggers from real earthquakes, we generated phone-triggers for a simulated network to test performance sensitivities of our network detector. We used a 1° by 1° box and randomly distributed N stations within the box where N can be 100, 200, 300, 400 or 500. We allowed randomly distributed false triggers at a rate based on the assumption that 10% of phones initially trigger due to movement every second, and then 7% are

classified erroneously as an earthquake (Movie S3). We then added earthquake triggers due to earthquakes using the following method (Movie S4).

The trigger time for each phone is based on Fig S5. Given the distance of the phone from the epicenter, the trigger time is randomly selected within the time range given by the blue lines on Fig S5. To determine a probability that a phone triggers, we developed a simple regression relation for the probability of a trigger given the estimated peak ground acceleration (PGA) at the site. We estimated peak ground acceleration at the site using a standard ground motion prediction equation (38). Our observations from the M5.1 La Habra earthquake are that the probability a phone triggers is 1, 0.8, 0.4, 0.25, 0.1, 0.01 at epicentral distances up to 5, 10, 20, 30, 40, and 50 km respectively. Using these observations we performed a simple regression between $\log_{10}PGA$ and trigger probability. The resulting regression relation is

$$P = 0.798 \times \log_{10}PGA - 0.557$$

where P is the probability that a phone is triggered. In the case that $P > 1$ we set $P = 1$ and $P < 0$ we set $P = 0$.

In 1000 simulations for each value of N, there were no false network earthquake detections. For N=500, 400 or 300 the performance is similar with all events detected ~3.5 sec after the origin time with location errors of ~4km (Table S3, Movie S4). For N=200, 32 of the 1000 events were not detected, and 11 were not detected for N=100 (Table S3). It also took longer to detect the events, and the locations had larger errors for N=100 and 200 illustrating the need for a dense distribution of smartphone detectors for this approach to work. The N=300 case corresponds to average distance between phones of 6.4 km. We also did 1000 simulations with only noise data without earthquakes, and found the algorithm did not have false alert issued. This is due to the requirement that >60% of active phones trigger within a 10 km radius for an earthquake to be declared.

4 Estimate warning time for Katmandu, Nepal

For the M7.8, 25 April 2015 earthquake in Nepal we can estimate the possible warning time in Katmandu using our smartphone seismic network approach. The location of the epicenter is 28.147°N, 84.708°E, and the location of Katmandu is 27.700°N, 85.333°E, a separation of 79 km. The S phase of the earthquake will arrive at Katmandu in 25.2 seconds based on iasp91 model (39). Assuming there are smartphones near the location of the earthquake, and because our network detection algorithm makes use of phones within 10 km of the epicenter, we would expect an earthquake detected when the S-wave reaches 10 km from the epicenter, which is 3.9 seconds after the origin time based on iasp91. Therefore, there could be ~20 seconds warning if we have a smartphone seismic network in Nepal.

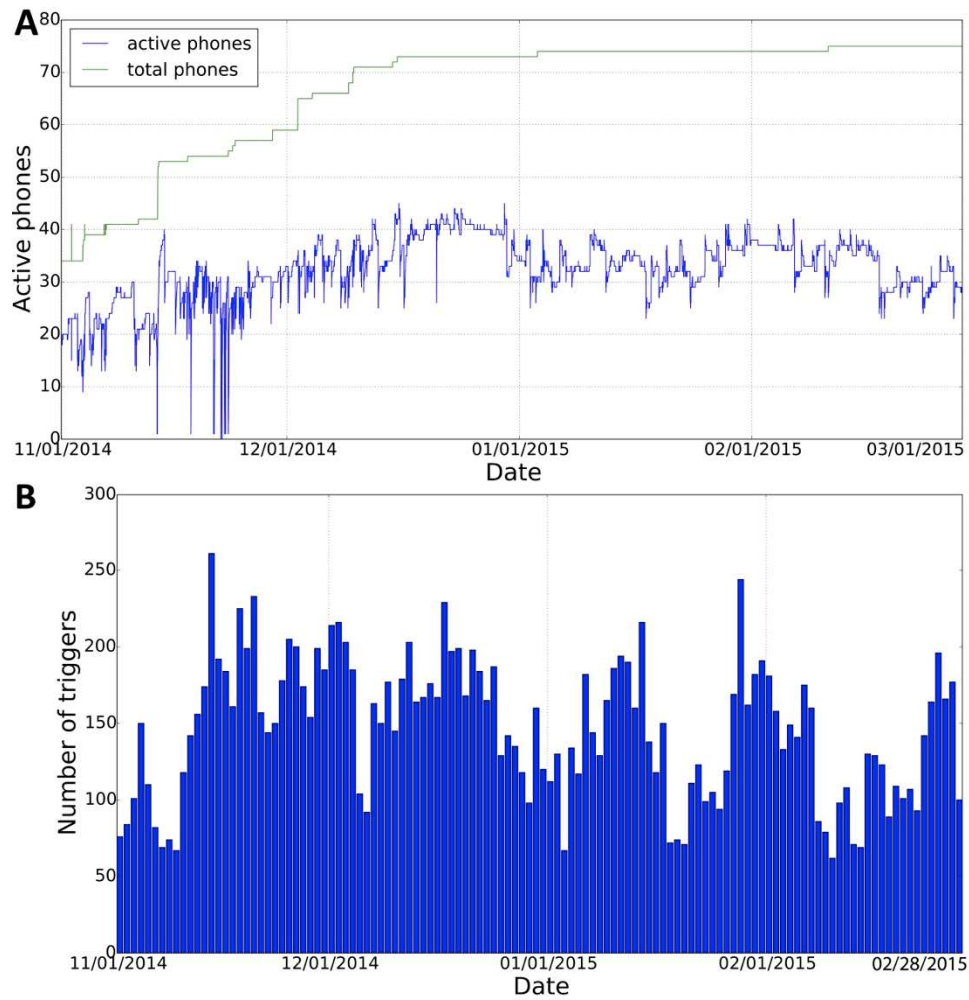


Fig. S1. MyShake activity 1 November 2014 to 28 February 2015. (a) Number of phones that downloaded MyShake and registered with our network (green curve), and the number of active phones running the application on a given day based on the SOH information (blue curve). Server at CPC restarts during the first month is the reason the number of active phones drops to zero. (b) Number of phone triggers each day with waveforms uploaded to the CPC, a total of 17600 triggers were collected.

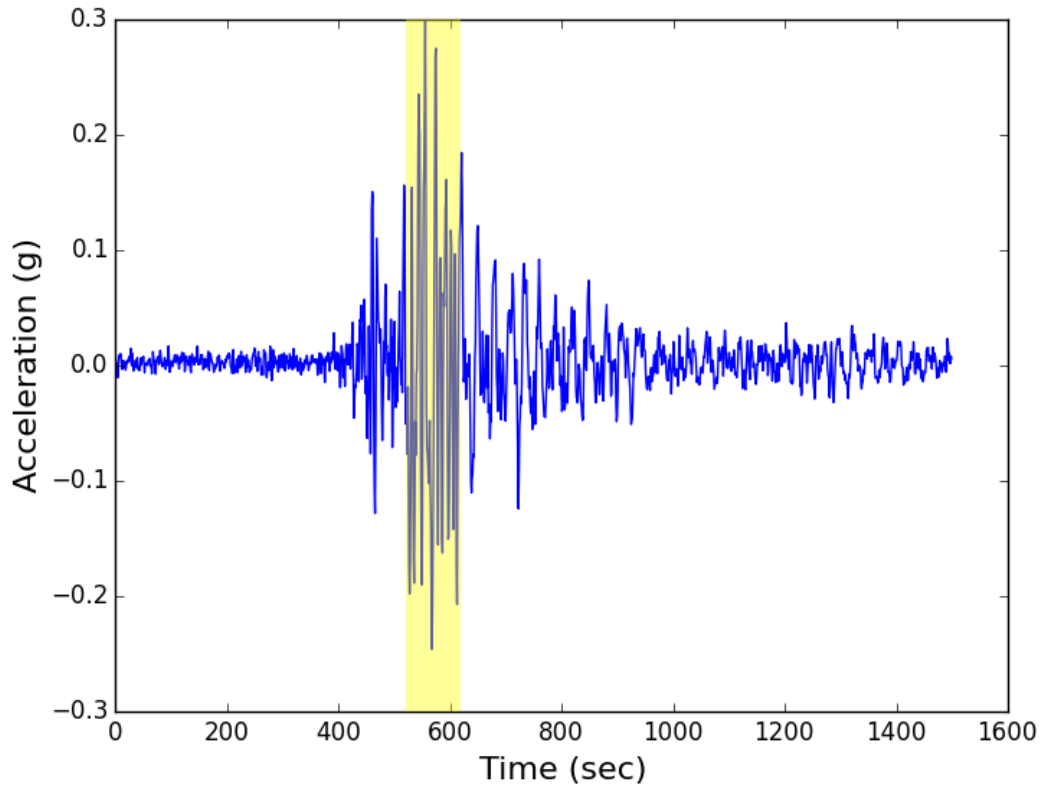


Fig. S2. Example earthquake record used to train the ANN classifier algorithm. The waveform is the EW component from a regional network station 16.5 km from the epicenter of the western Tottori earthquake (M7.3) of October 6, 2000. The data has been modified to represent a smartphone recording at the same location. Only 2-sec windows of data from the yellow region were used to train our algorithm.

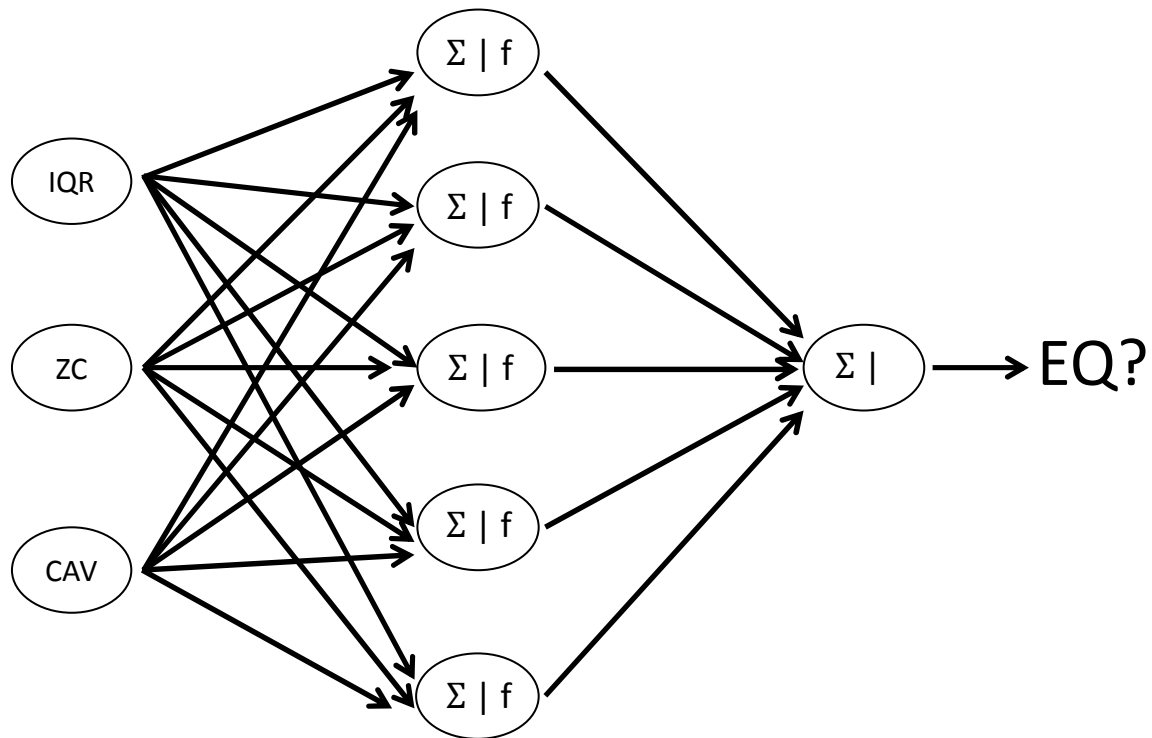


Fig. S3. Structure of ANN classifier algorithm. It has three layers: one input layer with 3 nodes, a hidden layer with 5 nodes, and an output layer with 1 node. For the hidden layer and output layer, the inputs from the previous layer to the each node will be first summed and then fed into an activation function shown as f .

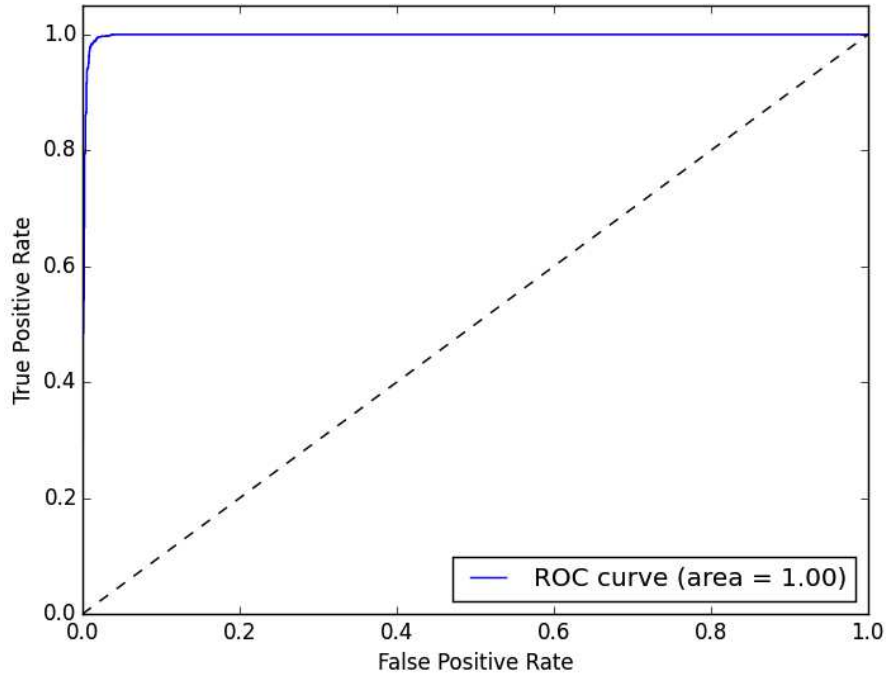


Fig. S4. Receiver operating characteristic curve. Shows the ANN classifier performance on 30% test data split from the training data. The ROC curve shows the false positive rate (classified as earthquake when it is a non-earthquake) on the x-axis, against true positive rate (classified as an earthquake when it is an earthquake) on the y-axis. Ideally, the curve will climb quickly toward the top-left corner meaning the model correctly predicted the cases. Our result is quite close to the ideal cases.

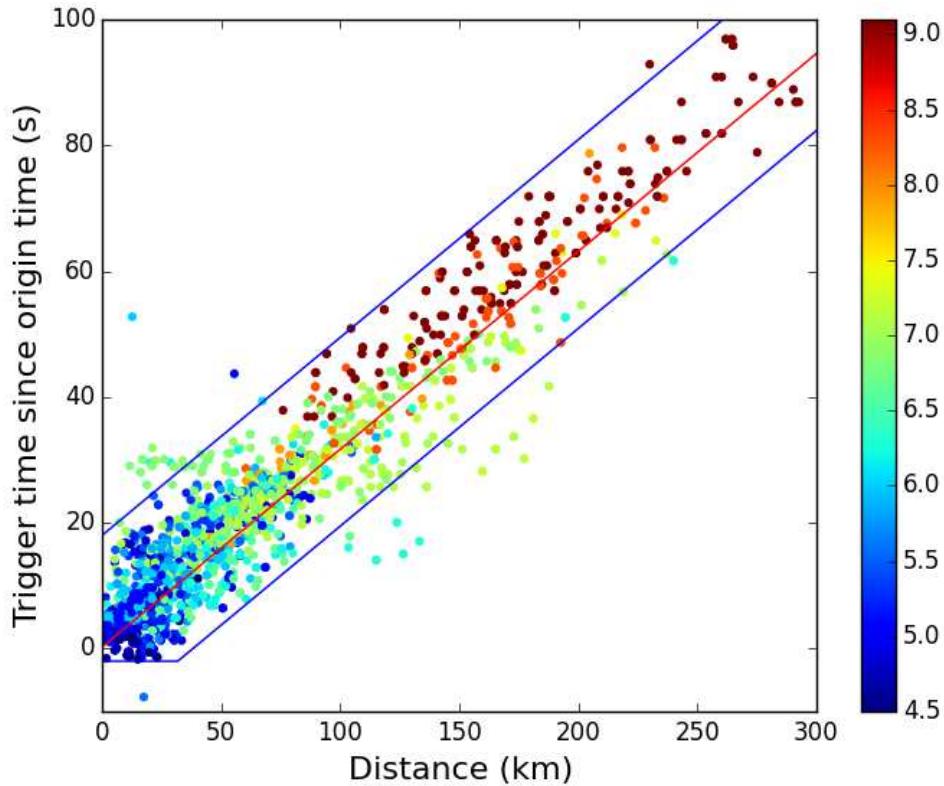


Figure S5. Phone trigger times versus epicentral distance. The regional network data from California and Japan was modified to phone-quality data and then our classifier applied to the data to determine when a trigger occurs. The red line is the best-fit to the data and has a moveout velocity of 3.2 km/sec; most triggers are generated by the S-wave or the later surface wave. The blue outline is the time-space window used for association of triggers with an event by the network detection algorithm.

Table S1. Accuracy score for ANN classifier with 10-fold cross-validation. The score row shows the accuracy score for each run defined as

$$accuracy(y, \hat{y}) = \frac{1}{n} \sum_{i=0}^{n-1} I(\hat{y}_i = y_i),$$

where n is the number of samples used, and I is a function that takes 1 when the argument is true, and 0 otherwise. This means that if the ANN classifier correctly classify the data, then I will be 1, otherwise 0. So the higher the average score, the better the ANN classifier. We ran a 10-fold cross validation, which means we split data into 10 sets of $n/10$ and trained on 9 datasets and tested on 1 dataset. We repeat this process 10 times, and each time select a different dataset as test set. The mean row shows the average score from the 10 runs, with the deviation showing in the parentheses.

| | | | | | | | | | | |
|-------|-----------------------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Score | 0.9893 | 0.9830 | 0.9839 | 0.9811 | 0.9919 | 0.9919 | 0.9893 | 0.9857 | 0.9821 | 0.9966 |
| Mean | 0.986 (± 0.001) | | | | | | | | | |

Table S2. Simulated network detection performance for U.S. earthquakes. Simulated phone triggers from 2014 La Habra M5 earthquake and 2004 M6 Parkfield earthquake were used to test the network detection algorithm. The magnitude, location and origin time estimates and errors are given for the initial MyShake estimates.

| Earthquake | Origin time | Event latitude | Event longitude | Alert time | Mag. | Location error (km) | Origin time error (sec) |
|----------------------|----------------------------|----------------|-----------------|------------|------|---------------------|-------------------------|
| La Habra: True | March 29, 2014 04:09:42 | 33.932 | -117.917 | | 5.1 | 3.76 | 2 |
| La Habra: Estimated | 04:09:44 | 33.900 | -117.930 | 04:09:47 | 5.2 | | |
| Parkfield: True | Sep 28, 2004 17:15:24 | 35.815 | -120.374 | | 6.0 | 1.55 | 2 |
| Parkfield: Estimated | 17:15:26 | 35.810 | -120.390 | 17:15:28 | 5.5 | | |

Table S3. Simulated network performance for various phone densities. N is the number of randomly distributed stations within a 1°x1° box (~111x111 km); we did 1000 simulations in each case for a M6.0 earthquake. The location errors are the differences between the true earthquake location and the estimated earthquake location. The origin time errors are the time difference between the true earthquake origin time and that estimated. The detection time is the time after the true earthquake time that the algorithm detects it. In all cases we show the average value \pm standard deviation. The last column shows the number of simulations in which the earthquake was not detected.

| Number of stations | Location error (km) | Origin time error (sec) | Detection time after true origin time (sec) | Events not detected (out of 1000) |
|--------------------|---------------------|-------------------------|---|-----------------------------------|
| N = 100 | 14.02 \pm 8.92 | 4.41 \pm 2.80 | 6.59 \pm 2.87 | 11 |
| N = 200 | 5.29 \pm 4.42 | 1.77 \pm 0.96 | 3.93 \pm 0.99 | 32 |
| N = 300 | 4.36 \pm 4.79 | 1.42 \pm 0.77 | 3.53 \pm 0.80 | 0 |
| N = 400 | 3.56 \pm 3.18 | 1.27 \pm 0.66 | 3.48 \pm 0.69 | 0 |
| N = 500 | 3.50 \pm 3.86 | 1.26 \pm 0.73 | 3.51 \pm 0.63 | 0 |

Movie S1. Network detection animation for the 2014 M5.1 La Habra earthquake. Animation shows snapshots of trigger detections every second for the 2014 La Habra earthquake simulation. Time is shown in seconds after the event origin time. Grey dots are stations; pink indicates a trigger. The true earthquake location is the red star and grey circles indicate 10, 20 and 30 km radius. The blue star represents the estimated event location, first detected at 5 sec. This simulation used regional network data for this event modified to phone-quality records. The phone trigger and classifier algorithm is then applied to generate phone-triggers; the network detection algorithm then detects the event.

Movie S2. Network detection animation for the 2004 M6.0 Parkfield earthquake. Same format as Movie S1.

Movie S3. Simulation of network detection with no earthquake. This movie shows simulation with only smartphone triggers due to human motions that are erroneously classified as an earthquake. The active phones are shown as grey dots, when they trigger they turn pink. They remain pink for 20 sec, which is the duration of time the trigger remains in the network-detection buffer. The grey circle has a 10 km radius; 4 triggers must occur within a 10 km circle to initiate a possible event declaration. No false earthquake detections occur.

Movie S4. Simulation of network detection for an M6 earthquake. Grey dots are stations; pink indicates a trigger. The animation starts 1 sec before the earthquake to show the noise triggers already in our 20 sec association buffer. The red star is the true earthquake location with grey circles at 10, 20 and 30 km. The algorithm detects the earthquake 3 sec after the origin and the estimated location is shown as a blue star. This is one of the 1000 simulations with N=400 stations.