

A Preliminary Result of Implementing a Deep Learning-Based Earthquake Early Warning System in Italy

Abiodun Adebowale[1(✉)], Federica Di Michele[2,3], and Bruno Rubino[1]

[1] Department of Information Engineering, Computer Science and Mathematics, University of L'Aquila via Vetoio, loc. Coppito, I, L'Aquila, Italy
abiodunsaheed.adebowale@student.univaq.it, bruno.rubino@univaq.it
[2] INGV, via Alfonso Corti 12, Milan, Italy
federica.dimichele@ingv.it
[3] GSSI, via M. Iacobucci 2, L'Aquila, Italy
federica.dimichele@gssi.it

Abstract. In this paper we present the preliminary results of a study using a deep-learning tool named LSTM (Long Short-Term Memory) network, to classify seismic events as near-source and far-source, with the final purpose of developing efficient earthquake early warning systems. We use a similar approach as in [15], applied to a database, named Instance, containing information about 54,008 earthquakes that occurred in Italy. Although these are preliminary results, the method shows a good ability to detect far-source events with an accuracy of about 67%. For near-source events, the method shows an improvable result with an accuracy of 57%.

Keywords: Earthquake Early Warning system (EEWs) · Long Short-Term Memory (LSTM) · Seismic Events

1 Introduction

It is well known that earthquakes hit without warning, often causing victims and extensive damage to buildings and infrastructures. They cannot be predicted, however, once an earthquake occurs and is detected it is possible to provide some advance notice of ground shaking, from a few seconds to a few minutes depending on the size of the earthquake and the location of the site. This idea has led to the development of *Earthquake Early Warning systems (EEWs)*. Many countries including Mexico, Japan, Turkey, Romania, China, Italy, and Taiwan have all implemented EEWs to identify earthquakes quickly and monitor their development to provide timely alerts in advance.

In the 1990s, the first automated earthquake pre-detection systems were constructed in Calistoga, California. The system automatically activates a citywide siren to inform everyone in the region of an earthquake [20], In 2006, Japan

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
O. Gervasi et al. (Eds.): ICCSA 2023, LNCS 13956, pp. 744–756, 2023.
https://doi.org/10.1007/978-3-031-36805-9_49

began using its EEWs in real-world situations. In order to alert the public, a warning system was put in place on October 1, 2007. The Urgent Earthquake Detection and Alarm System (UrEDAS) is meant to allow automated braking of bullet trains in case of a dangerous earthquake event [2,3,14]. The ShakeAlert early warning system was built and activated in Vancouver, British Columbia, Canada in 2009. The George Massey Tunnel, which runs under the Fraser River, is a vital piece of transportation infrastructure, and its protection was a top priority. The mechanism is programmed to shut the tunnel entry gates automatically in the case of an impending hazardous seismic event [25]. In 2018, the *ShakeAlertLA* app became the first EEWs to be made accessible to the general public to alert the public of shaking in the Los Angeles area, and in 2019, the state of California implemented a statewide alert distribution system using mobile applications and the Wireless Emergency Alerts (WEA) system, [9,17,23]. ShakeAlert system works as follows: both P (or compressional) and S (or shear) waves are generated during a seismic event. The quickest p-wave triggers sensors in the terrain, sending data to a ShakeAlert processing center to detect the earthquake's location, magnitude, and shaking. USGS issues an alert if the earthquake meets the hazard requirements. ShakeAlert partners may utilize the message to inform individuals to "Drop", "Cover", and "Hold On" or initiate an automated action [12]. Depending on the local faults and the available ground motion data, several EEWs may be necessary to quickly detect a seismic wave [12].

Many interesting results are available in the literature concerning the mathematical tool used for EEWs. Yamada et al. [24] outlined a process for determining whether seismic data are near or far from the point of origin using Fisher's linear discriminant analysis and two Bayesian model class selections to find the best combination of the peak ground-motion parameters. They reported that at 20 seconds, the probability of being in the near-source at 13 stations is computed to be greater than 50%. Kuyuk et al. [16] studied a Japan earthquake 7.3 Mw using a continuous wavelet transform approach to estimate the nearest and farthest stations to the epicenter.

More recently, artificial intelligence techniques such as deep learning have been utilized. The potential use of deep learning algorithms in seismology has been explored for purpose of classification, estimation, and interpretation of many geophysical phenomena such as volcanic eruptions and earthquakes. Several prior studies in seismology utilized Artificial Neural Network (ANN) for capturing pertinent features of seismic data. For example, Bose et al. [6] developed an ANN-based approach to EEWs for finite faults. They developed a shallow neural network model with two layers to estimate the hypocenter location, magnitude, and the progress of the expansion of an ongoing earthquake.

Anggraini et al. [4] developed a deep learning method for classifying p-wave and noise data and using them to estimate earthquake locations for EEWs in Indonesia. Only earthquakes in a cluster near to the station's hypocenter are considered for feature selection in the waveform, thus resulting in a high rate of false alarms. Apraini et al. [5] used Random Forest, a deep-learning method to

estimate and classify the magnitude of earthquakes for EEWs. In order to achieve a high level of performance in the specified earthquake parameters, Saad et al. [18] suggested a technique based on a convolutional neural network (CNN) that can extract meaningful characteristics from waveforms. The proposed method achieves classification accuracies of 93.67%, 89.55%, 92.54%, and 89.50%, respectively, for magnitude, origin time, depth, and location.

The aim of this paper is to provide a mechanism for separating near-source and far-source earthquakes in the shortest possible time from the arrival of the p-wave, following the idea first developed by [15]. A database specifically designed for Italy, named Instance [19], is used to test the tools.

With respect to [15] we improved the latency and minimized the computational resources by choosing the right activation functions which reduce the time spent training the model. In more detail, [15] used Softmax activation function which required more epochs (1400) to train, which in turn increases the time required to train the model. In this work, we used Sigmoid activation function since we are dealing with a binary classification problem, hence only 360 epochs are required to fully train the model, which is significantly faster.

2 Materials and Methods

When trying to figure out how to design an intelligent system, it proves relevant to study the structure of the brain and use it as a model to follow. This is the inspiration behind the creation of an ANN, a subset of machine learning which then found a major development in deep-learning algorithms.

The functional unit of the human brain is the neuron. The biological neuron cell has a nucleus and many dendrites and one long axon. The axon's length may range from a few times to tens of thousands of times longer than the cell body. Near its terminus, the axon divides into several branches known as *telodendria*, and at the tips of these branches are minute structures known as synaptic terminals (nerve endings), which are linked to the dendrites or cell bodies of neighboring neurons. In a similar way, an artificial neuron receives information in the form of an input vector, then there is a nucleus with elaborated signals and an output layer. Roughly speaking, an ANN is composed of many neurons connected together. In particular, a neural network contains three kinds of layers, an input layer, one (or more) hidden layers, and an output layer.

To analyze a time series, as in our case, a particular type of Artificial Neural Network called Recurrent Neural Networks (RNNs), can be employed. RNNs use sequential or time-series data as input. They are built to have the ability to recall knowledge from the past and apply it to new situations. They are differentiated by their *memory* since they are built to have the ability to recall knowledge from the past and apply it to new situations. In other words, instead of being independent of the inputs and outputs, RNNs are reliant on the previous parts in the sequence. Depending on the structure of our input and output data, we can categorize sequence modeling as [22]:

- **Many-to-one**: The output is a fixed-size vector, not a sequence, and the incoming data is a sequence. For instance, the classification of seismic wave characteristics, where the inputs are a sequence of ground motion data and the output is a classification label.
- **One-to-many**: The output is a sequence, and the incoming data is in a standard format.
- **Many-to-many**: Both input and output data are streams of sequences. An example is language translation, where a data stream from one language is the input sequence, and a data stream, translated into another language, is the output sequence.

To overcome the limit of vanishing gradient and effectively learn long-term dependencies in sequential data, a new algorithm named Long Short-Term Memory (LSTM) has been introduced in [13].

LSTM is a type of neural network that uses a series of memory cells (also called "blocks") to store information over time. Each cell contains a hidden state vector, h_t, and a memory cell vector, c_t, which are updated at each time step t based on the input, x_t, and the previous state, h_{t-1} and c_{t-1}. The update equations for the memory cell and hidden state vectors are defined in Algorithm 1. where Sigmoid is the sigmoid activation function, tanh is the hyperbolic tangent activation function, $\mathbf{W}$ are weight matrices, and $\mathbf{b}$ are bias vectors. The gates in the above equations (f_t, i_t, and o_t) allow the network to selectively control the flow of information through the memory cells, enabling it to forget old information, store new information, and output relevant information. This makes LSTM particularly useful for tasks that involve long-term dependencies and sequential data. Although the LSTM cell appears just like a conventional cell, it has two separate states: h and c (the letter c stands for "cell"). Short-term and long-term states may be thought of as separate entities representing h and c respectively [10]. A brief representation of the algorithm used is given below.

2.1 Algorithm Behind LSTM

Algorithm 1: The LSTM Algorithm

Input: $\mathbf{n}$, the length of the sequence
for $i = 1$ **to** n **do**
  if $i == 0$ **then**
    $\mathbf{h}^{(t-1)} \leftarrow$ small random number
    $\mathbf{C}^{(t-1)} \leftarrow$ small random number
  else
    $\mathbf{h}^{(t-1)} \leftarrow \mathbf{h}^{(t)}$
    $\mathbf{C}^{(t-1)} \leftarrow \mathbf{C}^{(t)}$
  end
  $SigmoidControllers \leftarrow (\mathbf{f}, \mathbf{i}, \mathbf{o})$
  for σ; **in** $SigmoidControllers$ **do**
    $\sigma_t \leftarrow \mathbf{sigmoid}\left(\mathbf{W}_{x\sigma}\mathbf{x}^{(t)} + \mathbf{W}_{h\sigma}\mathbf{h}^{(t-1)} + \mathbf{b}_\sigma\right)$
  end
  $\mathbf{g}_t \leftarrow \mathbf{tanh}\left(\mathbf{W}_{xg}\mathbf{x}^{(t)} + \mathbf{W}_{hg}\mathbf{h}^{(t-1)} + \mathbf{b}_g\right)$
  $\mathbf{C}^{(t)} \leftarrow \left(\mathbf{C}^{(t-1)} \odot \mathbf{f}_t\right) \bigoplus (\mathbf{i}_t \odot \mathbf{g}_t)$
  $\mathbf{h}^{(t)} \leftarrow \mathbf{o}_t \bigoplus \mathbf{tanh}\left(\mathbf{C}^{(t)}\right)$
end

In preparing the data and building the deep learning model, the following Python packages were used:

- **ObsPy**: The ObsPy project is an open-source initiative that aims to provide a Python framework for the analysis of seismological data. It offers parsers for the most popular file formats, clients that can connect to data centres, and seismological signal processing algorithms that enable the manipulation of seismological time series. [21]
- **Keras**: Keras is a high-level Deep Learning API that makes it easy to train and execute neural networks. It may be built on top of TensorFlow, Theano, or Microsoft Cognitive Toolkit (previously known as CNTK). TensorFlow has its own version of this API, named **tf.keras**, which supports several advanced TensorFlow features (e.g., the ability to efficiently load data). [10]
- **Tensorflow**: TensorFlow is an advanced library for distributed numerical computing. It makes it feasible to train and operate extremely large neural networks effectively by dividing the computations among possibly hundreds of multi-GPU machines. TensorFlow (TF) was developed at Google and is used in many of the company's large-scale Machine Learning projects. Open source was released in November 2015.

3 Results and Discussion

As mentioned in the introduction, the aim of this work is to adapt the ANN-based approach proposed by [15] in order to apply it to Italy. Most of the data we use comes from the databases of the Istituto Nazionale di Geofisica e Vulcanologia (INGV) and in particular from the database *INSTANCE - The Italian seismic dataset for machine learning*. The recorded waveforms span both weak (HH, EH channels) and strong motion (HN channels), each waveform trace is recorded at 100 Hz, and presented in both counts and ground motion physical units, with the vast majority of the data coming from the Italian National Seismic Network (network code IV) (Fig. 1).[1]

The dataset collected from Italy consisted of 1,159,249 three-component waveforms representing weak and strong motion data in the north-south, east-west, and up-down directions. Subsequently, the dataset was reduced to 10,863 three-component waveforms of strong motion data and 6,040 three-component waveforms after balancing across each magnitude from 3 to 7. Figure 2a illustrates the geographical locations and depths of all seismic events in the database, while Fig. 2b displays the positions of recording stations. Similarly, Figs. 3a and 3b depict the earthquake hypocenters and recording station positions after applying a filter to include only earthquakes within a 50 km epicentral distance in the Italian dataset.

We have processed and filtered this data in a similar fashion as in [15]. We divided the data from the seismographs for near-source and far-source earthquakes by defining a near-source earthquake as an earthquake with a distance

[1] http://doi.org/10.13127/instance.

from the epicenter below 17 km. Far-source earthquakes are defined by a distance above 17 km and below 50 km from the epicenter. Both classes are of a minimum magnitude of 3 on the Richter scale. This is an arbitrary choice and other approaches are possible and will be described and analyzed in the future.

Furthermore, the dataset is then balanced across all magnitudes using the SMOTE method [7] (Table 1).

Table 1. Table showing the difference between near- and far-source stations defined by epicentral and hypocentral distances.

Near Source vs Far Source		
Strong-motion Stations (HN)	Epicentral distances (km)	Hypocentral distances (km)
7495191.IV.T1244..HN	15.39	18.086
1736871.IV.STAL..HN	10.85	15.099
10655971.IV.FEMA..HN	5.44	9.674
12697681.IV.T1218..HN	19.317	20.986
4326711.IV.CDCA..HN	41.752	43.789
22823821.ST.RONC..HN	41.756	43.105

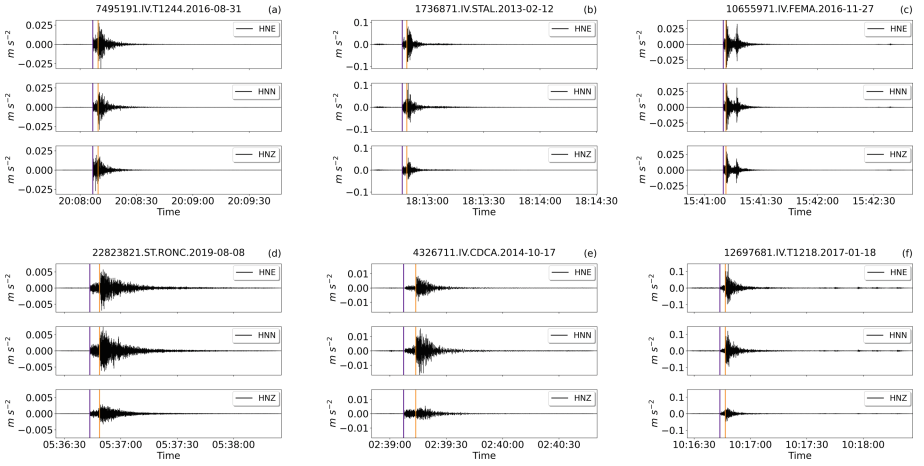
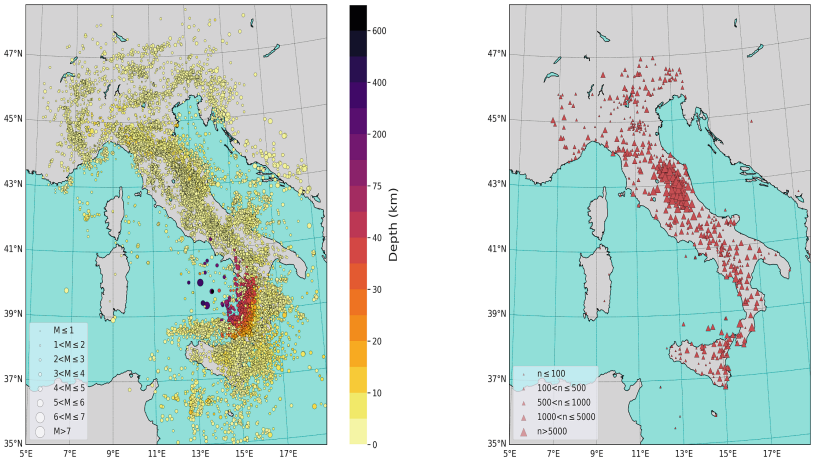
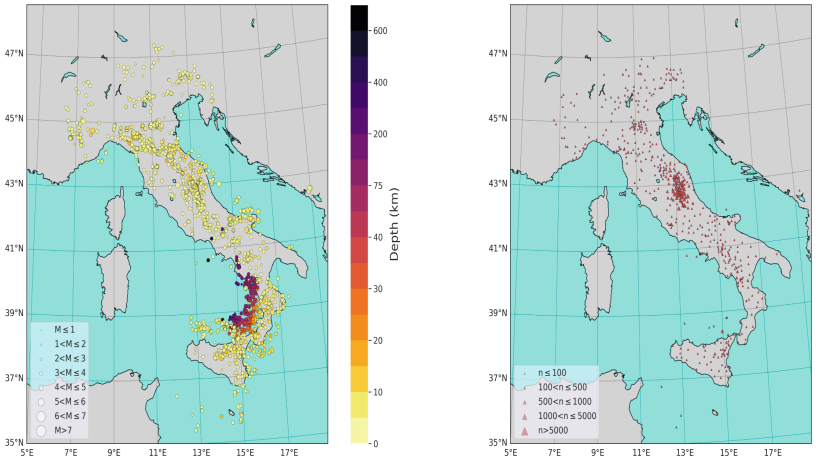


Fig. 1. Sample earthquake waveforms from INSTANCE broadband HN channels, selected at random. Three-component traces are inserted into each row at random so that the first row contains near-source events, while the second row contains far-source events. The arrival time of primary and secondary waves are shown with blue and orange marks respectively. (Color figure online)



(a) Hypocenter depth of full dataset (b) Map of stations in the full dataset

Fig. 2. Earthquakes in the dataset are shown on a color-coded map as solid circles of varying depths and the geographical locations of the stations that contributed to all the events.



(a) Hypocenter depth within 50km (b) Map of stations

Fig. 3. Earthquakes in the dataset are shown on a color-coded map as solid circles of varying depths and the geographical locations of the stations that contributed to the filtered events.

Similarly to [15], we selected six fundamental features of any seismological data, namely the time series data extracted from one second of p-wave (which is the three components accelerations (NS, EW, UD)) and the filtered acceleration for each of the components. Since the ground motion from near- and far-sources

tends to have different amplitudes, we combined their absolute and cumulative sums. The frequency range of high-frequency vibrations is most prominent at close proximity to the source. The amplitude of high-frequency ground motion decreases more quickly with distance than low-frequency vibration. Since high-frequency vibrations have a stronger correlation with epicentral distance, they may serve as a useful metric for separating stations with close and remote sources [11,24].

Adding the absolute and cumulative sums increased our features from 6 to 12. We also added the geometric mean of the waveforms as in [15], which makes our features a total of 13, with each feature having 100 data points 100 Hz frequency of one second of p-wave data. Therefore, the input matrix for each station is 13 by 100. Figures 4 and 5 show the time series plot of one second of p-wave data of the features and the normalized values respectively. The effects of ground vibrations diminish with increasing distance, as shown in Fig. 4. Indeed the near-source amplitude in blue is much higher than the far-source in red, this is a powerful characteristic of the p-waves that can be used by the LSTM model to detect near and far coming waves.

Looking solely at the overall form of the time series in Fig. 5, it is difficult to tell the different series apart. An earthquake's effects are most pronounced near the epicenter, for instance the near-source motion is stronger around the epicenter and weakens with increasing distance compared to the far-source motion. The final dataset comprises of 6040 three components waveforms of strong motion data and balanced sets across each magnitude from 3 to 7, with each event containing 13 features namely: strong-motion acceleration, filtered acceleration, absolute acceleration, and cumulative absolute acceleration of the three strong-motion components (**NS, EW, UD**), including the geometric mean. Each of the 13 features contains 100 time-series data points. This makes our input shape $6040 \times 13 \times 100$ normalized acceleration data points. Amongst them, 4976 are used to train the model and 1244 to test it.

The first layer of our model is the Sequential model. For neural networks with a single stack of layers linked sequentially, this is the most basic *Keras* model. A Sequential Application Programming Interface is being used here. Next, we added an LSTM layer, consisting of 100 neurons. A dropout layer of 0.2 and finally, since this is a classification problem, we used a Dense output layer consisting of a single neuron activated by a sigmoid activation function to forecast zero or one for each of the problem's two classes (near-source and far-source).

A log loss (binary cross-entropy in *Keras*) is utilized as the loss function since this is a binary classification task. We used the powerful ADAM optimization method. Due to rapid overfitting, the model is only fit for 400 epochs. In order to spread out the training data evenly, we employed a large batch size of 32. Different parameters were used in the hyperparameter tuning, such as using *EarlyStopping* callback to prevent training resource wastage, different activation functions, and varying learning rate, which had little effect on the final output. As a result, we reached the limit of what can be achieved in terms of precision. This demonstrates that we have hit the limits of possible optimization. Our result is similar

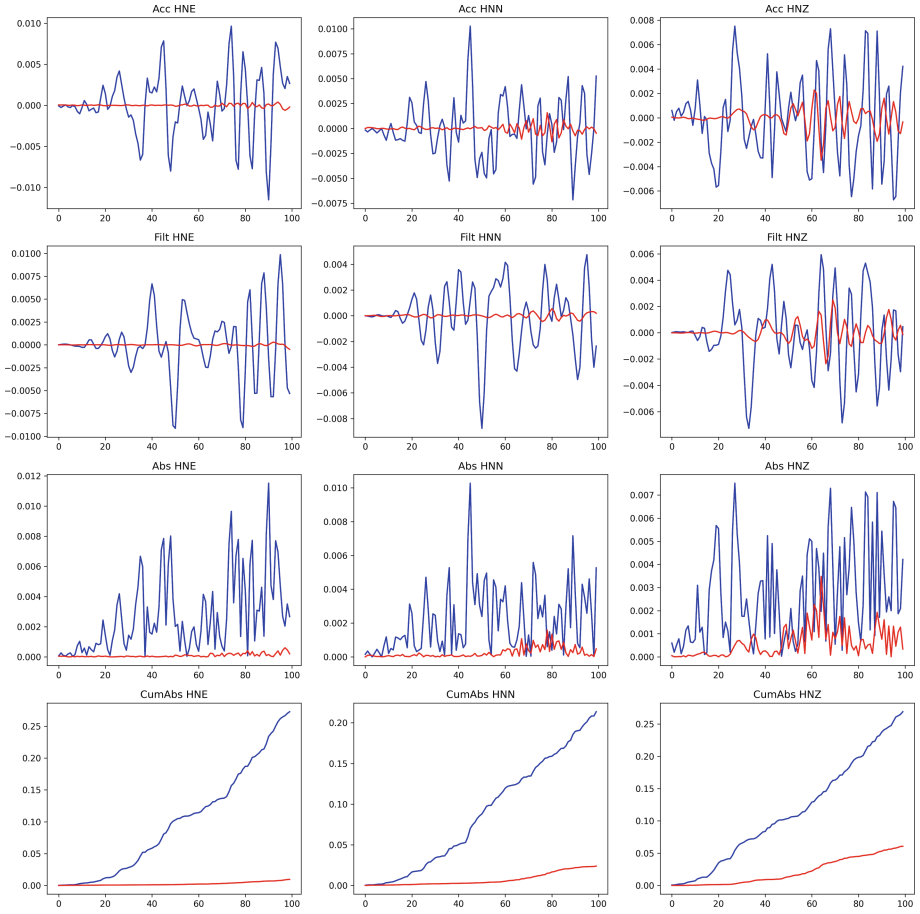


Fig. 4. Features Plot of One Second of p-wave data. The near-source amplitude in blue is much higher than the far-source in red. Evidence from the figures indicates that p-wave is amplitude dependent (Color figure online)

to what was achieved in [15] with a similar dataset but different neural networks tools. There are recordings from 2488 near- and 2488 far-source events included in the training set. Out of 2488 near-source records, 152 do not have the near-source label. The model firstly trained on the training set, reaches an accuracy of 94%. However, 3 events of the far-source data are incorrectly categorized. A total of 97% rate of accuracy is achieved. Only 155 of the 4976 events have the wrong label. Less satisfactory but still acceptable and in line with the available literature are the results achieved by applying the trained model to the test dataset. In more detail, the method has a good ability to detect far-source events with an accuracy of about 67% for the training dataset. Worse is the performance for near-source with an accuracy of 57% (Tables 2 and 3).

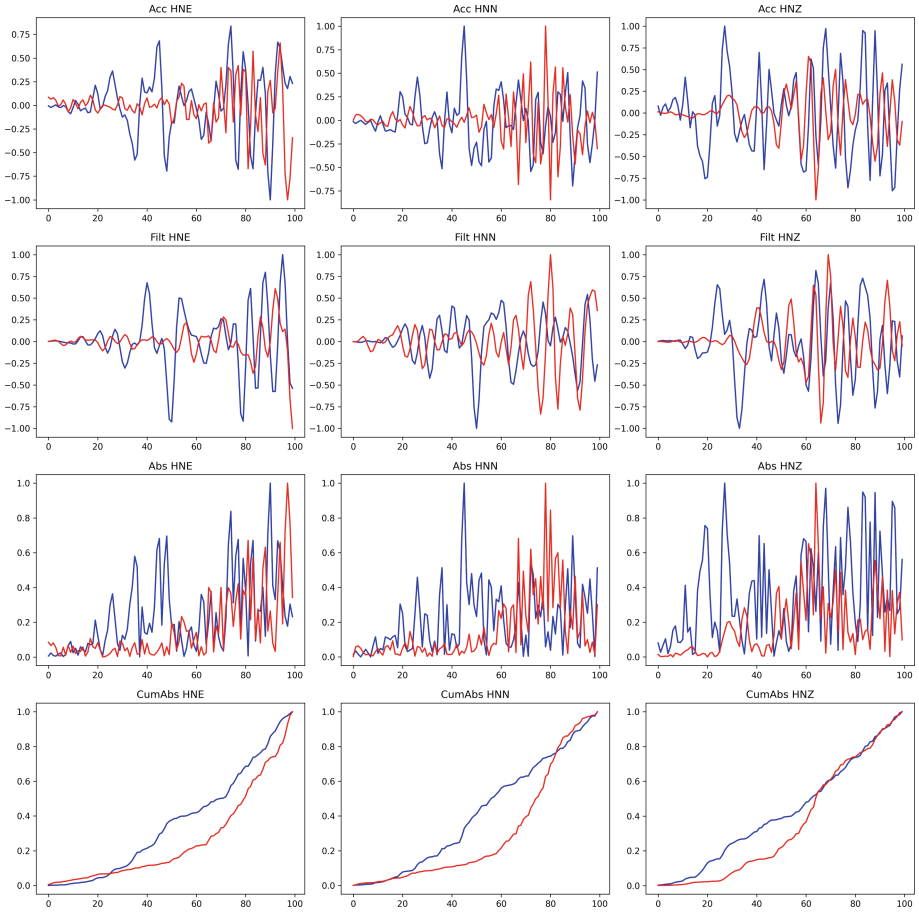


Fig. 5. Normalised Features Plot of One Second of p-wave data

Table 2. Confusion Matrix reporting the result of the LSTM algorithm applied to the training dataset. Accuracy is very good, especially for far-source events where it reaches 99%.

Training Performance of LSTM model on Italian dataset				
	Predicted Right	Predicted Wrong	Total	Accuracy
Near Source	2336	152	2488	94%
Far Source	2485	3	2488	99%
Total Events	4821	155	4976	97%

Table 3. Confusion Matrix reporting the result of the LSTM algorithm applied to the test dataset. Accuracy quite good, especially for far-source events where it reaches 67%, but less satisfactory for near-source events (accuracy 57%)

Test Performance of LSTM model on Italian dataset				
	Predicted Right	Predicted Wrong	Total	Accuracy
Near Source	354	268	622	57%
Far Source	417	205	622	67%
Total Events	771	473	1244	62%

4 Conclusions

In this paper we report preliminary results of a study employing deep learning techniques to classify earthquakes as near- and far-source, only one second after p-wave arrival, using the INSTANCE dataset, developed for Italy in [19]. We used the same approach proposed in [15], modifying the activation function from Softmax to Sigmoid, which significantly improved the computational resources on training the model. [15] used Softmax activation function which required more epochs (1400) to train with Tensorflow, which in turn increased the time required to train the model. In this work, we used sigmoid activation function since we are dealing with a binary classification problem, hence only 360 epochs were required to fully train the model making it significantly faster, as described in [1]. Our results are quite satisfactory for earthquakes classified as far-source, in fact, in this case, the accuracy reaches 67%, which is in line with the available literature. The performance on the near-source, however, is less satisfactory (57% of accuracy) and can certainly be improved. Below are some possible improvements that will be the subject of future studies.

The model can be improved by

- changing the boundary between the near- and far-source, which is fixed at 17 km in this work.
- improving the constraints on the data set.
- increasing the number of events considered, extending the study to the entire Mediterranean area, or including data from numerical simulation. [8]

References

1. Adebowale, A.: A deep learning-based technique to classify near-source and far-source earthquakes. Master's thesis, University of L'Aquila, L'Aquila IT (2022)
2. Agency, J.M.: Overview of the earthquake early warning system (2008). http://www.seisvol.kishou.go.jp/eq/EEW/kaisetsu/eew_naiyou.html
3. Agency, J.M.: What is an earthquake early warning? (2008). https://www.jma.go.jp/jma/en/Activities/eew1.html
4. Anggraini, S., Wijaya, S.: Daryono: Earthquake detection and location for earthquake early warning using deep learning. J. Phys. Conf. Ser. **1951**, 012056 (2021). https://doi.org/10.1088/1742-6596/1951/1/012056

5. Apriani, M., Wijaya, S.: Daryono: Earthquake magnitude estimation based on machine learning: Application to earthquake early warning system. J. Phys. Conf. Ser. **1951**, 012057 (2021). https://doi.org/10.1088/1742-6596/1951/1/012057
6. Böse, M., Wenzel, F., Erdik, M.: Preseis: A neural network-based approach to earthquake early warning for finite faults. Bull. Seismol. Soc. Am. **98** (2008). https://doi.org/10.1785/0120070002
7. Chawla, N., Bowyer, K., Hall, L., Kegelmeyer, W.: Smote: Synthetic minority oversampling technique. J. Artif. Intell. Res. (JAIR) **16**, 321–357 (2002). https://doi.org/10.1613/jair.953
8. Di Michele, F., et al.: Spectral element numerical simulation of the 2009 l'aquila earthquake on a detailed reconstructed domain. Geophys. J. Int. **230**(1), 29–49 (2022)
9. Government, C.: Wireless emergency alerts (2019). https://earthquake.ca.gov/wireless-emergency-alerts/
10. Géron, A.: Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly, 2nd edn. (2019). ISBN: 978-1-492-03264-9
11. Hanks, T.C., McGuire, R.K.: The character of high-frequency strong ground motion. Bull. Seismol. Soc. Am. **71**(6), 2071–2095 (1981). https://doi.org/10.1785/BSSA0710062071
12. Hazards, E.: Earthquake early warning around the world. https://www.usgs.gov/programs/earthquake-hazards/science/earthquake-early-warning-around-world
13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
14. Kumagai, J.: A brief history of earthquake warnings (2007). https://archive.ph/20130415005659/http://www.spectrum.ieee.org/print/5156
15. Kuyuk, H., Susumu, O.: Real-time classification of earthquake using deep learning. Procedia Comput. Sci. **140**, 298–305 (2018). https://doi.org/10.1016/j.procs.2018.10.316
16. Kuyuk, H., Talha, K., Yildirim, E., Sümer, Y.: Assessment of near- and far- field earthquake ground motion with wavelet transform. In: Conference: Symposium on Modern Methods in Science (BMYS2010) (2010)
17. Lin II, R.G.: Long-awaited earthquake early warning app for l.a. can now be downloaded (2019). https://www.latimes.com/local/lanow/la-me-ln-earthquake-early-warning-app-20190102-story.html
18. M. Saad, O., Hafez, A., Soliman, M.: Deep learning approach for earthquake parameters classification in earthquake early warning system. IEEE Geoscience and Remote Sensing Letters, pp. 1–5 (2020). https://doi.org/10.1109/LGRS.2020.2998580
19. Michelini, A., Cianetti, S., Gaviano, S., Giunchi, C., Jozinović, D., Lauciani, V.: Instance - the italian seismic dataset for machine learning. Earth Syst. Sci. Data **13**(12), 5509–5544 (2021). https://doi.org/10.5194/essd-13-5509-2021
20. Podger, P.J.: Calistoga to get an earful of nation's first quake siren (2001). https://web.archive.org/web/20140223054531/http://members.napanet.net/
21. Python framework for processing seismological data: Obspy. https://docs.obspy.org/index.html
22. Sebastian, R., Vahid, M.: Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn and TensorFlow, 2nd edn. Packt Publishing Ltd., Birmingham (2017). ISBN: 978-1-78712-593-3
23. Snibbe, K.: California's earthquake early warning system is now statewide (2019). https://www.mercurynews.com/2019/10/15/what-you-should-and-should-not-do-during-an-earthquake/

24. Yamada, M., Heaton, T., Beck, J.: Real-time estimation of fault rupture extent using near-source versus far-source classification. Bull. Seismol. Soc. Am. **97**(6), 1890–1910 (2007). https://doi.org/10.1785/0120060243
25. Zaicenco, A., Weir-Jones, I.: Lessons learned from operating an on-site earthquake early warning system. In: 15th Proceedings of World Conference on Earthquake Engineering (2012)