Predict passive seismic events with a convolutional neural network

Hanchen Wang¹, Tariq Alkhalifah¹ & Qi Hao².
1. King Abdullah University of Science and Technology 2. King Fahd University of Petroleum and Minerals

SUMMARY

The ample size of time lapse data requires tremendous event detection and source locating capabilities, especially in areas like shale gas exploration regions where a large number of passive seismic events are often recorded. In many cases, the real-time monitoring and locating of these events are essential to production decisions. Conventional methods face considerable drawbacks. For example, traveltime based methods require travelt ime picking; migration methods, on the other hand, require many wavefield modeling applications. These human interaction based pickings or wavefield simulations face severe issues when too many passive sources need to be located, which is common in shale gas explorations. A purely automatic method with no human interactions and less computational cost is necessary. Recently, machine learning has been utilized for this task, whether to identity passive seismic events, or to locate their sources once they are identified and picked. We propose to use an artificial neural network to directly map seismic data, without any picking, to locations of potential seismic activities. A visual geometry group (VGG) neural network is trained on synthetic acoustic data, corresponding to known sources, to provide predictions of new passive source locations, and other source features such as source wavelet peak frequencies, amplitudes and the number of sources within the data segment. To reduce the size of the input-to-the-network seismic data, we correlate the traces with the central trace to allow the network to focus on the curvature of the input data. We train the network to handle both single and double events that might be included in the correlation window. An initial application of the approach on a simple V(z) model demonstrates its potential.

INTRODUCTION

Hydraulic fracturing is one of the commonest used methods in oil and gas extraction, especially in those reservoirs with dense rocks such as shale. Usually, water or other liquids are injected into the reservoir area to create high pressure conditions, which eventually crack the dense rock in order to make the oil and gas flow more freely. The injection process often ignites micro-seismic events, which can be monitored through sensors in a monitoring well or on the Earth’s surface. Through the monitoring, the locations of the passive seismic events are usually estimated, which in turn can help us monitor the fracturing process. Such information helps the engineers optimize their injection strategy. Hence, locating the passive event is important to the hydraulic fracturing process.

During the fracturing monitoring, thousands of passive seismic events can be recorded. Locating all the events in a short time is a huge challenge. Conventional ways to locate the sources include methods that are based on traveltime picks of both P-wave and S-wave arrivals (Eissner et al., 2009). These methods are fast computational wise but they suffer from manual efforts of picking the arrivals. Besides, the errors in the picking lead to mislocation of passive sources. Some other methods are based on migration techniques, utilizing the whole recorded data. Usually, the recorded data are back-propagated into a given model from their receiver locations and the source locations are determined by a proper imaging condition (Artman et al., 2010; Wang and Alkhalifah, 2018a). Another category of passive imaging methods is based on full waveform inversion (FWI). The inversion is also responsible for velocity model building using a proper objective of data matching (Sun et al., 2016; Wang and Alkhalifah, 2018b). However, both migration based and FWI based methods require large computational time for simulating the wavefields, because there are tens of thousand of passive events recorded during one single hydraulic fracturing process.

Recent studies in machine learning provide new approaches to overcome the issue. Since they are purely automatic without human interactions and do not require wavefield simulations, they can greatly reduce human and computational costs. Several techniques have already been proposed on automatic earthquake localization. With the explosion in computational resources, machine learning algorithms to detect, locate and interpret passive seismic events automatically are gaining a lot of attention. A convolutional neural network (CNN) is trained using more than two thousand human induced passive seismic events that are recorded by two stations to roughly estimate the source locations (Perol et al., 2018). Kriegerowski et al. (2018) also trained a CNN using more than two thousand events from West Bohemia, recorded on nine local stations, to locate clustered earthquakes precisely.

In this abstract, a visual geometry group (VGG) neural network is trained using synthetic P-wave data. Once the VGG network is well trained, it can be used for real-time locations of recorded passive seismic events. Besides the source locations, we also train the network to predict other source features such as source wavelet frequencies, amplitudes and number of sources in one single data segment. We demonstrate the proposed method with a synthetic dataset and we also add random noise to the data in order to show the noise sensitivity of the method. The synthetic test shows convincing results.

THEORY

Machine learning is a branch of artificial intelligence (AI) function that imitates the workings of the human brain in processing data and creating patterns for the use in decision making. Deep learning is a subset of machine learning utilizes networks capable of learning supervised (or unsupervised) from data that are labeled (or unlabeled). The VGG neural network is one of the well known deep learning neural networks, which was developed recently (Simonyan and Zisserman, 2014). Compar-
pared to deep learning architectures with linear layers only, the main concept of a VGG network is stacking of convolutional layers to create deep neural networks. The detailed architecture of the VGG network we use in this abstract is shown in Figure 1. Several convolutional layers are stacked together as one convolutional feature (or block). Inside the convolutional layers, a filter is convolved with the input data to capture the unseen features of the input data. The output values of the convolutional layers are passed through an activation function and down pooled by a max-pooling layer. Then the down pooled values are fed to the convolutional layers in the next block. After passing all the convolutional blocks, the values are passed to linear layers as inputs to generate the final predictions for the current epoch. Using the predictions, we calculate the loss function and back-propagate the loss to train the network with a certain learning rate. Repeat this training process and after sufficient epochs, the network will eventually learn the connections between the input data and their corresponding output labels. Then we can use the trained network to predict the similar labels of new data.

For the seismic event location problem, the input of the network is a correlated segment of the data from the monitoring system and the output layer will contain labels corresponding to event position (x & z). Inside the convolutional layers, a filter is convolved with the input data to capture the unseen features of the input data. The output values of the convolutional layers are passed through an activation function and down pooled by a max-pooling layer. Then the down pooled values are fed to the convolutional layers in the next block. After passing all the convolutional blocks, the values are passed to linear layers as inputs to generate the final predictions for the current epoch. Using the predictions, we calculate the loss function and back-propagate the loss to train the network with a certain learning rate. Repeat this training process and after sufficient epochs, the network will eventually learn the connections between the input data and their corresponding output labels. Then we can use the trained network to predict the similar labels of new data.

For the seismic event location problem, the input of the network is a correlated segment of the data from the monitoring system and the output layer will contain labels corresponding to event position (x & z), source wavelet peak frequencies (F), amplitudes (A) and the number of the sources in the segment (N). Passive seismic data from the field are extremely long, sometimes for days. Common pre-processing of the data may have it cut into several data segments once an event is detected. Thus, we may have one or more events in each data segment. However, it would be still too large of data to be trained since we need thousands of training data. We here choose to process the data by a correlation step, which is given by the below:

$$D(x, \tau) = d_{ref}(t) * d(x,t),$$

(1)

where \(d(x,t)\) is the data segment, \(d_{ref}\) is a chosen reference trace from the data segment, * is the correlation operation and \(\tau\) is the lag of the correlation operation. The data before and after the correlation are shown in Figures 2a and 2b, respectively. Note that the events are pulled to approximately zero lag after the correlation operation and thus we can further cut the data segment with a smaller time window around the zero lag position. In this way, we may further reduce the input data size. If the raw data record, which contains many events, is processed by the same correlation operation, all the events will be pulled to near zero lag position. Only the cross-talk may have large lag times. Although, in this case, the events will overlap, we still can train the network to identify each of them. In other words, we may significantly reduce the uncut record from several hours long to several seconds, which is obviously a huge saving. The loss function is defined as:

$$L = \sum_{i=1}^{n} \left( |x_i - x_{i}^{\prime}|^2 + |z_i - z_{i}^{\prime}|^2 + |F_i - F_{i}^{\prime}|^2 + |A_i - A_{i}^{\prime}|^2 + |N_p - N_{p}^{\prime}| \right),$$

(2)

where the subscriptions p & t refer to the predicted and true values and i is the source number. The output labels x and z are the source coordinates, \(F\) is the peak frequency, \(A\) is the maximum amplitude and \(N\) is the total source number in the input data. We choose to train the network by predicting such output labels because each of them has its own signature in the input data. The x coordinate is driven by the event apex horizontal position; \(z\) is defined mainly by the moveout; \(F\) is defined by the thickness of the event; \(A\) is defined by the energy recorded; \(N\) is determined by how many events appear in the record. This loss function is back-propagated to train the neural network in order to minimize the loss function.

The convolutional layers and the linear layers are used to learn the nonlinear relationships between the input data and the output labels. Increasing the convolutional layers or the numbers of neurons in each layer may provide better performance on the training set. However, if there are too many neurons in the network, it may also lead to the problem of over-fitting. Thus, the network will perform badly on the testing data. Therefore, the choice of the network parameters is quite important for better performance.

**Training and testing datasets**

The training and testing datasets for our network are generated synthetically. To create the training dataset, we compute waveform records corresponding to sources distributed randomly through the whole velocity model. In our experiment, the velocity is linearly increased with depth \(V(z)\) model), ranging from \(2\ km/s\) to \(2.64\ km/s\). The model size is \(0.64\ by\ 0.64\ km\). The monitoring system is set to be every grid point on the Earth surface (total of 64 receivers). The source parameters are randomly selected, which means the source position \((0.05 - 0.63)km\), frequency \((5 - 15)Hz\), and amplitude \((0.5 - 1)\) are all random numbers within these ranges. For each recorded event, we choose the central trace as the reference trace and apply the correlation operation. We normalize each input dataset to keep the input and output values of the network within a range \((-1 to 1)\), otherwise the training will be unstable, which in turn makes the amplitudes not as easy to be predicted. Totally \(2666\) training data, including \(1333\) each for single and double event cases, and \(666\) testing data, including \(333\) each for single and double event cases, are generated for the experiment.

**EXAMPLES**

The architecture of the VGG network we used here, shown in Figure 1, has six convolutional blocks, in which two convolutional layers are stacked together and followed by an activation function (Tanh) and a down sampling layer (Max-pooling). The last convolutional block is connected to three linear layers and the last linear layer provides the output predictions. We choose the Tanh function as the activation function because Tanh outputs both positive and negative values, while the famous ReLu function only provides positive values. In seismic data, we have both positive and negative values. ReLu will drop the negative information in the data but Tanh will keep it, which may help the training accuracy.

**Training with single-source noise-free data**

We first test our method on single-source noise-free training and testing data. A sample training data is shown in Figure 2b. We use a total \(1333\) events to train the network. The loss curve with respect to epochs is shown in Figure 3. The loss function decreases to \(15\%\) after \(50000\) epochs. The smooth decrease is mainly due to the small learning rate \((0.01)\) necessary in or-
The training process is not only accurate, but also efficient. We are sensitive to random noise and it can solve multi-sources issues. This experiment further shows the method is insensitive to random noise and it can solve multi-sources issues. We show that the VGG neural network can be used to efficiently predict accurate passive source locations for micro-seismic monitoring. We train the network on correlated acoustic waveforms of a synthetic dataset. We show that the network can predict not only the source locations, but also the frequencies, amplitudes and number of sources in the data, which makes it more realistic and useful. The efficiency compared to conventional methods makes it cheaper and faster in real-time passive event locating tasks. Of course, we assume we know the velocity in the area, which can be provided by a well log. Moreover, if a sufficient amount of already located sources are available, we can use these historical data, as well, to train the network instead of training it with synthetic data, which will possibly relieve us from the need to have a velocity model. Besides, we show the network is insensitive to random noise, which makes it more stable when dealing with real data.

ACKNOWLEDGMENTS

We thank KAUST for sponsoring this research. We also thank to the team of SWAG for their help during the research. Special thanks to KAUST Supercomputing Laboratory for supporting the computing resource.
Figure 3: The loss curve as a function of epochs.

Figure 4: The loss curve with random noise multi-sources data as a function of epochs.

Figure 5: (a) 50 samples of predicted source locations (red) compared to true ones (blue). (b) Errors map between the predicted source locations and the true ones.

Figure 6: (Top) Predicted source peak frequencies (red) compared with the true frequencies (blue). (Bottom) The errors between the two frequency series.

Figure 7: (Top) Predicted source amplitudes (red) compared with true amplitudes (blue). (Bottom) The errors between the two amplitude series.

Figure 8: (a) 50 samples of multi-sources predicted locations (red) with random noise compare with the true locations (blue). (b) Errors map between the predicted and the true locations in km.

Figure 9: (Top) Predicted multi-sources peak frequencies (red) with random noise compare to true frequencies (blue). (Bottom) The errors between the two frequency series.

Figure 10: (Top) Predicted multi-sources amplitudes (red) with random noise compare to true amplitudes (blue). (Bottom) The errors between the two amplitude series.

Figure 11: Predicted source numbers for (a) single-source and (b) multi-sources cases.
REFERENCES


Wang, H., and T. Alkhalifah, 2018a, Microseismic imaging using a source function independent full waveform inversion method: Presented at the .