Traveltime computation using a supervised learning approach

Jubran Akram*, Daniel B. Peter†, David W. Eaton*, and Hongliang Zhang*

*University of Calgary, Alberta, Canada

†King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia

SUMMARY

In real-time microseismic monitoring, the ability to efficiently compute source-receiver traveltimes can help in significantly speeding up the model calibration and hypocenter determination processes, thus ensuring timely information about the subsurface fractures for use in effective decision making. Here, we present a supervised-learning based traveltime computation approach for layered 1D velocity models. First, we generate numerous synthetic traveltime examples from a combination of source locations and layered subsurface models, covering a broad range of realistic P-wave velocities (2500–5000 m/s). Next, we train a multi-layered feed-forward neural network using the training set containing source locations and velocities as input and traveltimes as corresponding labels. By doing so, we aim for a neural-network model that is trained only once and can be applied to a wide range of subsurface velocities as well as source-receiver positions to predict fast and accurate traveltimes. We apply the trained model on numerous test examples to validate the accuracy and speed of the proposed method. Based on the comparisons with acoustic finite-difference modeling and a ray-shooting method, we show that the trained model can provide faster and reasonably accurate traveltimes for any realistic model scenario within the trained velocity range.

INTRODUCTION

In earthquake seismology, source-receiver traveltimes are frequently used in the inversion for velocity model and hypocenter locations (Kim and Baag, 2002). These source-receiver traveltimes can be conveniently modeled using any of the available methods including ray shooting, ray bending, and techniques that either involve solving eikonal equation or finding minimum traveltime path on a grid of points (Moser, 1991, and references therein). However, both the speed and accuracy of any traveltime computation approach are important; especially in real-time microseismic monitoring applications which require reliable and timely information about fracture propagation for effective stage planning and decision making.

Recently, machine learning approaches have gained considerable popularity for their efficiency in solving numerous geophysical problems including event detection, arrival-time picking, and inversion of velocity model and hypocenter locations (e.g., Akram et al., 2017; Ross et al., 2018; Zhang et al., 2020). These approaches mainly belong to supervised and unsupervised learning types. A supervised learning algorithm requires the user to provide a training set containing many examples of input features and their corresponding output labels. After sufficient training, the algorithm learns the mapping between

inputs and output. It is then able to create a useful approximation of the output for the new unseen inputs. On the other hand, an unsupervised learning algorithm does not require an input-label training set. It can learn the hidden patterns and create a new representation of the data, which might be easier for humans to understand (Müller and Guido, 2016).

Previously, numerous studies have used machine learning approaches for traveltime computation. For example, Kononov et al. (2007) applied radial-basis function neural networks to calculate traveltimes for use in seismic migration. This approach uses the location of different source-receiver pairs as input and their corresponding traveltimes as labels in the training set, while the velocity model remains fixed during the network training. bin Waheed et al. (2020) used a physics-informed neural networks (PINN)-based algorithm for solving the anisotropic eikonal equation. In this approach, a multi-layered feed-forward neural network is trained for a particular source location for a given velocity model. For a new source location and a modified model, the weights and biases from previous training are updated at a relatively fast speed.

We present a novel supervised learning based approach for traveltime computation for 1D layered velocity models, which are still commonly used in many passive monitoring applications due to their computational simplicity. Unlike the aforementioned approaches, we vary both layer velocities as well as the source-receiver locations during the generation of the training set. By doing so, we aim for a neural-network model that is trained only once and can be applied to a wide range of subsurface velocities as well as source-receiver positions to predict traveltimes. First, we explain the training of a multi-layered feed-forward neural network using synthetic traveltime examples from a combination of velocity models and source locations. Next, we validate the accuracy and speed of the trained model to generate traveltimes for different combinations of subsurface velocities, using comparisons with the acoustic finite difference modeling and a ray-shooting method. Finally, we show that the predicted traveltimes from the trained model can be incorporated in model calibration and hypocenter determination processes for speed up.

THEORY

An artificial neural network (ANN) is a mathematical computing paradigm, inspired by biological neural systems, that is commonly used to learn patterns and relationships in data. The basic building block of an ANN is the neuron, which is a simple processing unit. In an ANN, many neurons are interconnected in a way that is specific to the choice of network topology (e.g., feed-forward, recurrent). A fully connected feed-forward neural network is one of the oldest and simplest

Supervised learning for traveltimes computation

network, in which all neurons are connected, and the activation flows from input layer to output layer, without any feedback loops or connection between multiple layers. Figure 1 shows a generalized sketch of a feed-forward network. The input features $\mathbf{x} = [x_1, \dots, x_m]$ are given in the input layer and the output $\mathbf{y}$ is obtained from the output layer. Between the input and output layers, there can be many intermediate or hidden layers. Each input x_i is multiplied by a weighting coefficient w_j before reaching a neuron in the next layer. The output a_j for each neuron is the weighted sum of all inputs plus a bias term b_j (McCormack, 1991; Bishop, 2006), such that

$$a_j^{(l)} = \sum_{i=1}^m w_{ji}^{(l)} x_i + b_j^{(l)}, \quad (1)$$

where superscript (l) indicates that the corresponding parameters represent layer l . The outputs a_j are then transformed using a differentiable, nonlinear activation function $h(\cdot)$ as follows:

$$z_j^{(l)} = h(a_j^{(l)}). \quad (2)$$

This is used as input for the next layer, such that the corresponding activations can be written as:

$$z_j^{(l+1)} = h(a_j^{(l+1)}) = h\left(\sum_{i=1}^{n_l} w_{ji}^{(l+1)} z_i^{(l)} + b_j^{(l+1)}\right), \quad (3)$$

where n_l represents the number of hidden units in layer l . This process continues until the output layer is reached. The entire process is then repeated for all examples in the training dataset. At the end of each epoch, the neural network's weights and biases are updated, typically using a backpropagation technique, which evaluates the gradient of the error function, $E(\mathbf{w})$, between the predicted (y_k) and the desired outputs (t_k), with respect to network parameters. For this purpose, an efficient backpropagation technique can be used to evaluate the gradient of the error function $E(\mathbf{w})$. Numerous optimization techniques are commonly used to find the optimal parameters that minimize a certain error function. These include gradient descent, stochastic gradient descent, adaptive moment estimation (Adam), Nestrov accelerated gradient, RMSprop and Adagrad (Ovcharenko et al., 2018, and references therein). In this paper, we use Adam to find the optimal weights, and rectified linear unit (ReLU) as the activation function for each hidden unit. ReLU is a very simple nonlinear function that has worked well in many existing studies. It is expressed as (Gulli and Pal, 2017):

$$h(x) = \max(0, x). \quad (4)$$

The steps to train a feed-forward neural network for the regression problem of traveltimes prediction are described as follows:

1. Data collection for training:

- Specify the number of layers in the velocity model, based on experience or prior knowledge of the subsurface geology.
- Generate a synthetic traveltimes dataset using N_v random velocity models and N_{xr} randomly placed

Figure 1: Example of a feed-forward neural network architecture, with many hidden layers. Input layer comprises of m feature nodes. Each node connection between layers has a specific weight as indicated with variable thickness line-arrows. The rectified linear unit (ReLU) is used as a nonlinear activation function for each hidden neuron.

source and receiver positions, that change in every iteration of the velocity model. The traveltimes are computed using a ray-shooting method.

2. Input feature selection:

Convert the inputs and out from the synthetic data generation step into the following features:

- Source-receiver offset
- Absolute difference in source receiver depths
- Effective thickness for the raypath in each layer. When a source or receiver is located within a layer, the effective thickness is the difference between source or receiver's depth and layer's top or bottom depth depending on the ray direction.
- Effective layer velocities.
- Presence of source in a layer (0 or 1).
- Presence of receiver in a layer (0 or 1).

3. Data normalization:

Scale data using the min-max scaler, such that each feature ranges between 0 and 1

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}, \quad (5)$$

where x_{min} and x_{max} are the minimum and maximum values of a feature.

Supervised learning for traveltimes

4. Network training:

- Divide the collected data into training, testing, and validation sets. The training set comprises of examples that are used to fit the model. The validation set comprises of examples that are used to provide an unbiased evaluation of the model fit while tuning hyperparameters. In contrast, examples in the testing set are completely unseen by the model during training. Therefore, these are used to provide an unbiased evaluation of the final model.
- Carefully select the total number of hidden layers and the corresponding number of units (neurons) for each layer.
- Find the best hyperparameters by monitoring the network's performance for a range of settings during the training stage.
- Use the trained neural network on new data to predict traveltimes.

NUMERICAL EXAMPLES

We use a 10-layer velocity model to generate synthetic data for the feed-forward network's training. The choice of number of layers in the velocity model is representative since many studies (e.g., Bardainne and Gaucher, 2010; Tan et al., 2018) have previously used 1D models comprising of ten or fewer number of layers in downhole microseismic data analysis. For this training set, we use 5000 different velocity models and 100 randomly distributed source-receiver positions for each model realization. Also, for each model, velocities were randomly assigned to layers from a range (2500 – 5000 m/s), which is also a commonly used P-wave velocity range in many existing studies.

The generated synthetic traveltimes data is transformed into input features as specified in the theory section. Before training, we divide the input feature data into 70% training, 15% validation and 15% testing sets. The performance of the feed-forward neural network is evaluated for different parameters and hyperparameter settings. Here, we show results only from a 3-layered feed-forward neural network. The number of neurons used in the hidden layers are 800, 400, and 100, respectively. We use Adam optimizer with a learning rate of 0.001 to train the feed-forward neural network. The trained model is then applied to different velocity models for traveltimes prediction.

Figure 2 shows the comparison of traveltimes from a ray-shooting method (Tian and Chen, 2005) and the trained neural network for three different subsurface models (Figure 2a, homogeneous velocity (3500 m/s); Figure 2b, velocity increasing with depth; Figure 2c, velocity changing randomly with depth). The root-mean-square (RMS) of traveltimes difference between these two techniques for these three cases are 0.27 ms, 0.27 ms, and 0.24 ms, respectively. This shows that the trained model can provide robust traveltimes computation for many different velocity distributions, thus minimizing the need to retrain the

model. Even though the neural network model was trained for 10-layer velocity models, we can use the same model for fewer number of velocities, as in Figure 2a, we assigned the same velocity to all ten layers. In terms of computational effort, the neural network (including the data preparation time) offers 2.3 times speed-up over the used ray-shooting approach, for this example.

Figure 2: Comparison of traveltimes from a ray shooting method (solid, black) and the trained neural network (dashed, red). a) Homogeneous model. b) Velocities increasing with depth. c) Velocity changing randomly with depth. The grid size is 500 m x 1000 m with grid spacing of 1 m. The source position is shown as star at (500 m, 250 m). The neural network was trained for 10 layers. For all three cases, the root-mean-square (RMS) difference in traveltimes is close to 0.25 ms.

To further assess the accuracy, we compare the predicted traveltimes from the neural network model (Figure 2) with 2D acoustic finite-difference waveform modeling results (Figures 3a-c). In this modeling example, we use the second and fourth-order finite-difference operators for time and space derivatives. The predicted traveltimes contour and the wavefront at 0.06 s show very good overlap in all three cases.

We can also incorporate the neural network-based traveltimes prediction within model calibration and hypocenter location workflows for speed up. Figure 4 shows an application of hypocenter determination in which we use the neural network-

Supervised learning for traveltimes computation

based predict traveltimes to generate high resolution look-up tables for use in a grid search algorithm. The hypocenter location is determined using the maximum likelihood point on the grid. In this example, we use 100 source points that were randomly generated around a fixed location (assumed to be calibration shot location). The model used to locate these events was calibrated using a particle swarm optimization (PSO; Akram, 2020) algorithm in which incorporating the neural network based traveltimes allowed using a much higher number of particles for parameter search. The comparison between the true and estimated locations show good correlation in the overall cluster shape.

Figure 3: Comparison of neural network-based traveltimes (dashed, red) and 2D acoustic finite-difference modeling snapshot at 0.06 s. a) Homogeneous velocity model (3500 m/s). b) Velocities increasing with depth. c) Velocity changing randomly with depth. The grid size for both finite-difference modeling and traveltimes prediction is 500 m x 1000 m with grid spacing of 1 m. The source position is shown as star at (500 m, 250 m). The neural network was trained for 10 layers.

FUTURE WORK AND CONCLUSIONS

We have presented a supervised-learning based approach for fast and robust computation of traveltimes in 1D layered ve-

Figure 4: A synthetic example in which neural network-based traveltimes were incorporated into model calibration and hypocenter determination algorithms. Top panel shows a high resolution likelihood distribution for hypocenter determination. The maximum likelihood point, which in this example is the same as true location of the microseismic event, is represented by a star on the grid. Bottom panel shows the true and estimated locations of 100 source points that were randomly generated around a fixed point used to calibrate the velocity model.

locity models. We trained a three-hidden-layered feed-forward neural network on 5000 different velocity models and 100 randomly generated source-receiver positions for each realization of the velocity model. The trained model is then applied to different velocity distributions as subsurface model for traveltimes prediction for any source-receiver pair. We have compared the predicted traveltimes with results from a ray shooting method as well as with 2D acoustic finite difference modeling. The comparisons showed that the proposed algorithm is robust and can predict fast and reasonably accurate traveltimes for many different velocity and source-receiver distributions, without the need of further training. We have also shown that a trained neural network model can be directly incorporated within a global optimization algorithm such as PSO or an exhaustive grid-search algorithm for speeding up model calibration and hypocenter location processes, that can be useful in real time monitoring. Future research will focus on applying this approach to a field microseismic dataset.

ACKNOWLEDGMENTS

This work was made possible in part through CFREF support for the Global Research Initiative at the University of Calgary. The authors would also like to thank King Abdullah University of Science and Technology (KAUST) for financial support.