Journal of Petroleum Exploration and Production Technology (2022) 12:21–34
https://doi.org/10.1007/s13202-021-01304-0

REVIEW PAPER-EXPLORATION GEOPHYSICS

Seismic velocity modeling in the digital transformation era: a review of the role of machine learning

Abdullah AlAli[1] · Fatai Anifowose[2]

Received: 11 October 2020 / Accepted: 15 September 2021 / Published online: 28 September 2021

Abstract
Seismic velocity modeling is a crucial step in seismic processing that enables the use of velocity information from both seismic and wells to map the depth and thickness of subsurface layers interpreted from seismic images. The velocity can be obtained in the form of normal moveout (NMO) velocity or by an inversion (optimization) process such as in full-waveform inversion (FWI). These methods have several limitations. These limitations include enormous time consumption in the case of NMO due to manual and heavy human involvement in the picking. As an optimization problem, it incurs high cost and suffers from nonlinearity issues. Researchers have proposed various machine learning (ML) techniques including unsupervised, supervised, and semi-supervised learning methods to model the velocity more efficiently. The focus of the studies is mostly to automate the NMO velocity picking, improve the convergence in FWI, and apply FWI using ML directly from the data. In the purview of the digital transformation roadmap of the petroleum industry, this paper presents a chronologic review of these studies, appraises the progress made so far, and concludes with a set of recommendations to overcome the prevailing challenges through the implementation of more advanced ML methodologies. We hope that this work will benefit experts, young professionals, and ML enthusiasts to help push forward their research efforts to achieving complete automation of the NMO velocity and further enhancing the performance of ML applications used in the FWI framework.

Keywords Machine learning · Digital transformation · Velocity analysis · NMO velocity · Velocity inversion · Full-waveform inversion

Introduction

The velocity of a medium is a crucial step to obtain an accurate seismic image. Inaccurate velocity typically leads to incorrect positioning of the reflectors and hence causes a discrepancy in the image. Many techniques have been developed over the years to ensure a good velocity estimate of the subsurface. A common practice in seismic processing is to obtain the normal moveout (NMO) velocity from the common midpoint (CMP) gathers. The NMO velocity flattens the hyperbola in the data. Velocity search by semblance analysis (Yilmaz 1987) is the most common method used to estimate NMO velocity. Since it depends on manually picking the maximum energy semblance of the stacking velocity, this technique requires extensive human intervention. Hence, it is time-consuming, especially for large 3D volumes. Besides that, NMO velocity is often not accurate as it is based on the assumption of lateral homogeneity and fails to estimate complex structures. It provides a smooth velocity courtesy of the interpolation between the CMP gathers.

Velocity inversion is often implemented to obtain a more accurate velocity model. Full-waveform inversion (Tarantola 1984) is a common inversion method that provides a high-resolution velocity model by minimizing the least square misfit between observed data from the field and the modeled version. It is an ill-posed nonlinear optimization problem. Starting from an initial velocity model such as an NMO velocity, FWI updates the velocity iteratively until it converges to hopefully global minimum. However, due to the lack of low frequencies in the data, it often converges to a local minimum, especially with a poor starting model. Regularizing the inversion with constraints and prior information

✉ Abdullah AlAli
abdullah.alali.1@kaust.edu.sa

[1] King Abdullah University of Science and Technology, Thuwal, Saudi Arabia

[2] Saudi Aramco, Dhahran, Saudi Arabia

has been proven to achieve better convergence (Asnaashari et al. 2013; Kalita et al. 2019).

Recently and due to the advancements in computational resources and the availability of state-of-the-art algorithms, there has been a wide interest in machine learning (ML) applications within the geoscience community. These applications include formation tops identification using unsupervised learning techniques (Xuan and Murphy 2007) and seismic facies analysis using artificial neural network (ANN) (Wrona et al. 2018) and formation tops identification using supervised learning techniques (Maniar et al. 2018). Others are fault detection (Xiong et al. 2018; Wu et al. 2019) and salt interpretation (Zeng et al. 2019) using convolutional neural networks (CNN), and seismic attributes selection using ANN (Qi et al. 2020). Additional contributions were made to overcome the limitations of the current velocity building techniques. Most of these efforts were aimed at automating the picking process in the semblance related approaches while assisting FWI by low-frequency extrapolation, gradient manipulation, and regularization by using ML to converge faster. Some other attempts have been made to estimate the NMO or FWI velocity directly from the prestack shots/CMP gathers.

In this paper, we track and review the progressive efforts in the development of velocity models from the traditional empirical and analytical approaches to the use of machine learning techniques. First, we present an overview of some of the ML techniques that were applied in the studies that we reviewed in this paper. These include unsupervised learning technique such as clustering, supervised learning techniques such as deep neural networks, and semi-supervised learning technique such as meta-learning. Then, we discuss the ML applications in estimating NMO velocity. After that, we review the ML applications in the more advanced velocity modeling method, FWI. Finally, we conclude by making recommendations for potential future applications in view of contributing to the digital transformation effort of the petroleum industry.

Common machine learning techniques applied in the reviewed literature

There are three basic machine learning paradigms, namely supervised, unsupervised, and reinforcement learning (Bishop 2006). In the case of supervised learning, a training set, *X*, is used in the model function, *f(X)*, to build and optimize its relationship with a known target, *T*. *T* could be composed of labels (in case of classification) or continuous values (in the case of regression). Using a metric (loss) to estimate the similarity between the model prediction, $T' = f(X)$, and the actual target, *T*, the model is optimally tuned to predict the desired output. This can be expressed mathematically by

$$\text{Min loss}\left(T', T\right) \tag{1}$$

The goal of the tuning process is to keep the loss within a certain threshold or as low as practically possible. Common examples of losses used in training include mean squared error for regression problem and cross entropy for classification. Updating the model during the tuning process is often performed by gradient descent and back-propagation methods. Examples of techniques utilizing this type of learning methods include support vector machine (SVM), decision trees, and artificial neural network (ANN). For the unsupervised techniques, a corresponding target value of *T* is not available for a set of inputs, *X*. Rather, the learning goal typically involves discovering optimal groups with similar features in the data (e.g., clustering methods), reducing the dimension of the data (e.g., principle component analysis) or density estimation. Reinforcement learning is learning actions through trial and error based on rewards and punishments system in which the learning algorithm seeks to maximize the reward. An analogy of this type of learning is teaching a dog to sit. We emulate a situation, and the dog will respond in various ways. The dog will be rewarded with food if he sits. Next time when we emulate the same situation, the dog will sit expecting the reward.

A more recent ML technique known as meta-learning or "learning to learn" can be classified as a semi-supervised learning (Maclaurin et al. 2015; Andrychowicz et al. 2016). In meta-learning, the goal is to find the optimal hyperparameters such as loss, learning rate, regularization parameter, initial weights, and activation functions. Depending on the problem, one may choose whichever hyperparameters to optimize as desired and these hyperparameters will be referred to as "meta-variable." A loss, referred to as meta-loss, measures the performance of the network or model in terms of the degree of similarities of the model prediction and the actual target values. The algorithm starts by running series of training instances using some ML models. The meta-loss measures how well the ML model has succeeded in predicting the target. It would then propagate the error to update the network parameters. This type of learning is often implemented in nested loops. An inner loop to perform several training steps and an outer loop to optimize the meta-variables. Therefore, meta-learning requires high computational cost compared to a normal training process. Another requirement is that the network parameter should have higher-order derivatives such as the "gradient of the gradient."

The following subsections discuss in more detail some of the techniques used by geoscientists in velocity modeling, specifically clustering methods and ANN.

K-means and DBSCAN clustering

Clustering, a type of unsupervised learning, is a technique that identifies groups (or clusters) with similar features in the data (Gan et al. 2007). The goal for clustering algorithm is to partition the dataset into clusters (i.e., groups). Researchers have suggested different methods to choose the optimal number of clusters (e.g., Maclaurin et al. 2015; Salvador and Chan 2004). The user often determines that by trial and error. Various clustering algorithms such as K-means, expectation maximization, density-based spatial clustering (DBSCAN), and fuzzy clustering have been developed (Bradley et al. 1998; Rokach and Maimon 2005; De Oliveira and Pedrycz 2007). This section focuses on only the algorithms for K-means clustering and DBSCAN since they are applied in velocity applications.

K-means, where K represents the number of clusters, is one of the most commonly used clustering methods (Bishop 2006). It is implemented in the following steps:

1. Initialize cluster centroids.
2. Assign the data points to the closest clusters based on their proximity to the centroids.
3. Update the cluster centroids based on the mean of the data points within the cluster.
4. Repeat steps 2 and 3 until the centroids converge, that is, when the difference between the new and the current centroids is zero or less than some tolerance.

Figure 1 shows an example of this iterative process with three clusters from input data (a) to the last iteration.

DBSCAN is a density-based clustering algorithm (Ahmed and Razak 2016). By defining a radius, ε, and a minimum number, *minPts*, of points, DBSCAN classifies the points into three categories:

1. Core points: the points that contain more than *minPts* within the radius, ε.
2. Border points: the points that contain less than *minPts* within the radius, ε, and they are neighbor of core points.
3. Noise (outlier) points: the points that belong to neither core points nor border points.

Figure 2 shows an example of this classification using three as minimum number of points (minPts).

The workflow for the method is shown in Fig. 3 and composed of the following steps:

1. Select random points that are not assigned to a cluster or noise.

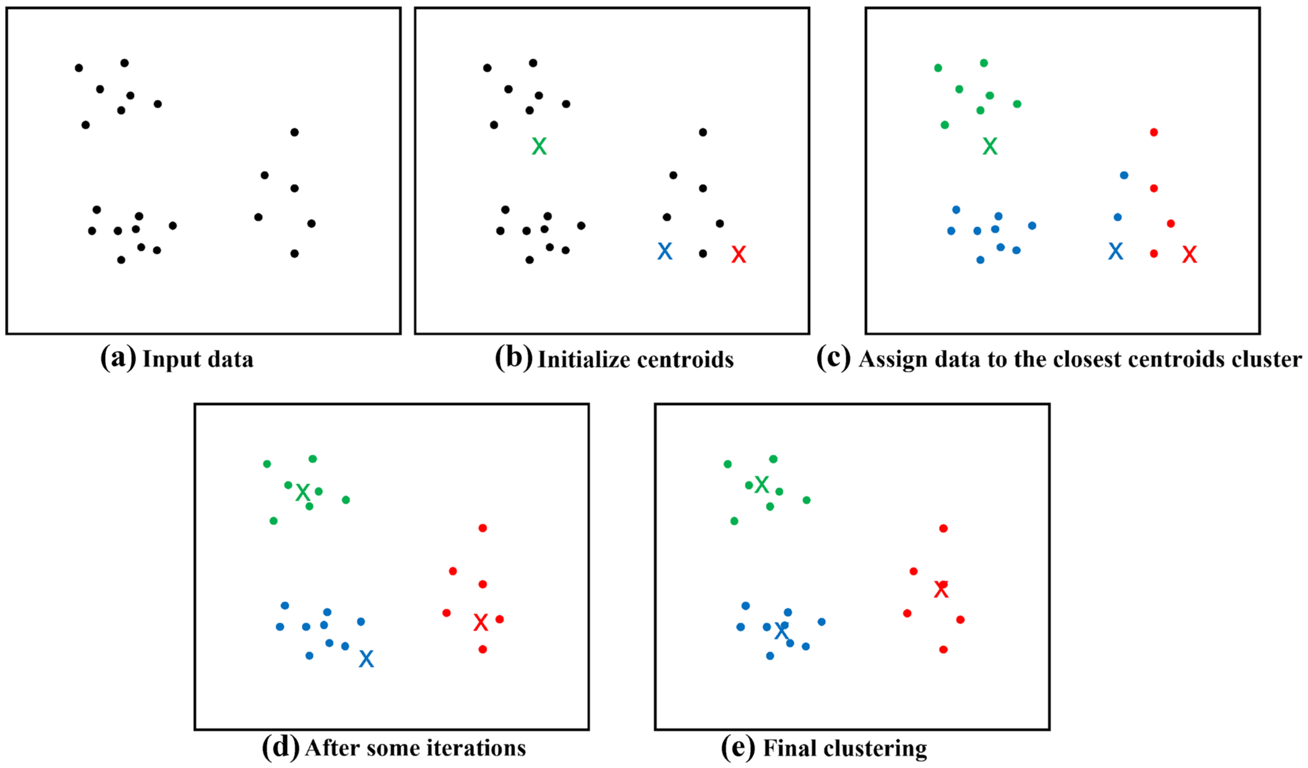
(a) Input data
(b) Initialize centroids
(c) Assign data to the closest centroids cluster
(d) After some iterations
(e) Final clustering

Fig. 1 *K*-means clustering process with $K=3$. '×' mark indicates the centroid and different colors indicate different clusters

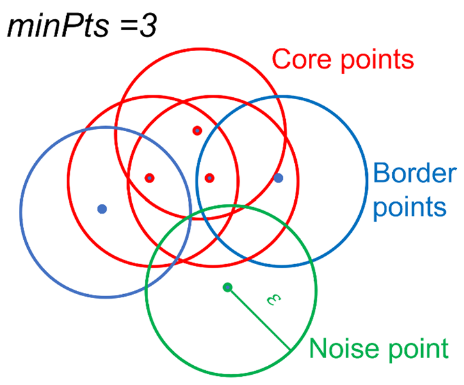

Fig. 2 Classification of core, border, and noise points with *minPts* = *3*. The red points indicate core points. The blue point is border point and the green point is noise

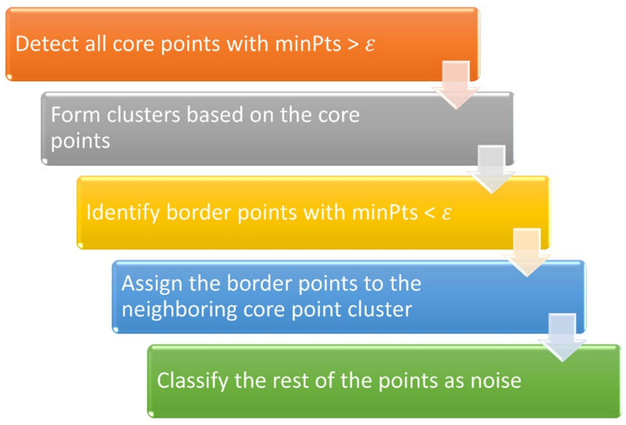
Detect all core points with minPts > ε
Form clusters based on the core points
Identify border points with minPts < ε
Assign the border points to the neighboring core point cluster
Classify the rest of the points as noise

Fig. 3 Flowchart for the DBSCAN clustering algorithm

2. Compute the neighborhood points within the distance, ε.
3. Assign the points using the following conditions:

   a. If the number of points within ε is larger than minPts, it becomes a core point.
   b. If the number of points within ε is less than minPts and it is in the neighborhood of a core point, it is a border point.
   c. Identify the point as noise if none of the above conditions is satisfied.

4. Assign the class of the core point to its border point.
5. The process is repeated until all data are assigned to a cluster or identified as outlier.

Artificial neural network

ANN is a powerful learning tool to approximate nonlinear functions. It is considered a supervised learning technique as it requires providing the true outcome for each point in the training data (Bishop 2006). The simplest form of the ANN is the fully connected layers (FC). FC is composed of three main layers: an input layer, which receives the input features; one or more hidden layers, which perform all the computations; and an output layer, which produces the final results. Each of the hidden layers consists of neurons, which are connected to the previous and next layers by weights. Inside a single neuron, the input vector is multiplied by the weights and a summation task is performed. To produce a final output of the neuron, an activation function is applied to the summation. Figure 4 shows a typical FC structure with three hidden layer for illustration.

Recently, more advanced ANN algorithms have been developed. These include convolutional neural network (CNN) (LeCun et al. 1995) and recurrent neural network (RNN) (Hochreiter and Schmidhuber 1997). CNN uses local convolutional filters to extract the spatial features from the inputs. It is widely used in image processing, object detection, image segmentation, and classification problems. RNN uses a memory variable that stores information from previous inputs in the new prediction. It is widely used for time series problems. Some of the common applications for RNN are natural language processing, language translation, and time series forecasting. RNN suffers from vanishing or exploding gradient problem. The structure of a version of the RNN algorithm, known as long short-term memory (LSTM), addressed this issue and has been commonly used (Hochreiter and Schmidhuber 1997).

Machine learning techniques applied to velocity estimation

This section discusses two major types of velocity models: NMO and velocity models obtained by FWI. The NMO velocity flattens the hyperbolas in the CMP gathers. NMO velocity is not accurate in most cases, as it is based on lateral homogeneity of the subsurface. Despite this, it is used for an

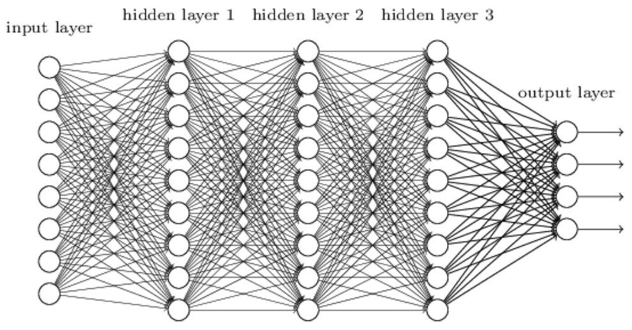

Fig. 4 Fully connected layers structures with three hidden layers. Each circle represents a single neuron

initial estimate for the subsurface model. FWI used iterative method to update the velocity model and obtain more accurate and higher resolution velocity. However, it should start from a good initial model. FWI is usually implemented by inverting for the low frequencies of the data and gradually progressing to the higher ones to ensure convergence. However, these low frequencies are often missing which make it challenging to FWI to converge to global minimum.

NMO velocity analysis

Traditionally, NMO velocity is obtained by creating semblance spectrum panels for several CMP gathers. The semblance panel consists of a range of velocities in one axis and the two-way travel time in the other. The highest energy semblance, which indicates the velocity at a particular time, is typically picked manually. With the help of machine learning, the picking process has been partially automated. Different strategies have been implemented with unsupervised learning such as clustering and supervised learning with ANN. Using ANN for automatic velocity picking is not new. Schmidt and Hadsell (1992) and Fish and Kusuma (1994) are few of the pioneers that used ANN for velocity analysis. The networks at that time were shallow and only able to extract some local information of the velocity semblance.

Several researchers suggested the use of K-means clustering as an auto-picking method for velocity analysis. Since seismic data are often contaminated with noise, Smith (2017) suggested using different attributes such as semblance, AVO auto-picking, and continuity of the gathers across offsets for clustering. This ensures more robustness. To further ensure that only the points around the high semblance are considered, Wei et al. (2018) proposed using some constraints such as adding a few manually picked semblances as a guide. Chen (2018) applied a threshold to keep the high-energy points. He implemented the K-means algorithm on the Gulf of Mexico (GOM) data. He achieved good picking except at some deeper parts where the spectrum is diffused. Bin Waheed et al. (2019) later compared the performance of K-means with the DBSCAN clustering algorithm. Their findings suggested that the lower value of K in the K-means algorithm would result in no picks, while the larger value is likely to lead to error (Fig. 5a). For DBSCAN, the values of the picks for the tested radii are similar as shown in Fig. 5b. They made a comparison between the picks for K-means ($K=5$), DBSCAN ($\varepsilon=0.02$) and the true velocity (Fig. 5c).

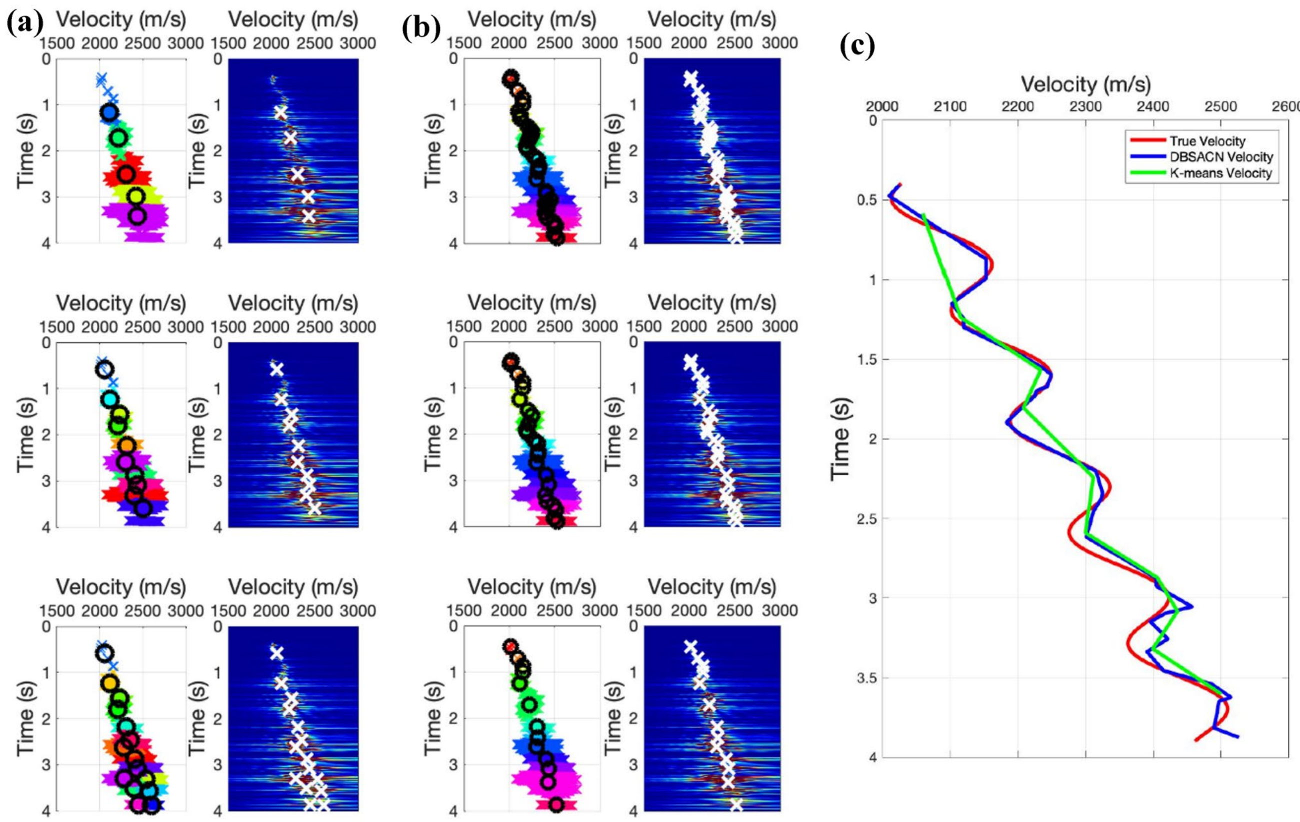

Fig. 5 **a** *k*-mean picking for $k=5$, 10 and 15 (from top to bottom). **b** is the DBSCAN picking for five minPts and radius 0f 0.02, 0.04 and 006. **c** is the comparison between *k*-mean with $k=5$, DBSCAN with radius 0.02 and the true velocity (bin Waheed et al. 2019)

Using an alternative approach, Ma et al. (2018) formulated the problem as a regression type rather than using the semblance picks. To achieve the regression objective, they used CNN to estimate the NMO velocity from the pre-stack CMP gathers directly. A predefined range of velocities were applied to the CMP gathers to flatten them. They trained the CNN model by taking mini-batches from the CMP gathers and outputting a number indicating the velocity errors. For example, the output is 1 if the CMP is flat, 0.9 for over-correction, and 1.1 for under-correction. The velocities corresponding to an output equal to 1 were selected for the velocity model. The method produced promising result when it was applied to the Marmousi model using velocities in the range of 0.9 and 1.1 of the true velocity.

Biswas et al. (2019b) suggested using RNN to obtain the velocity semblance picks from pre-stacked CMP gathers as a regression problem. The velocity governed the spread of the hyperbolas in time and offset. This means that the information needed to estimate the velocity at a particular time step is the neighboring temporal and spatial information. Therefore, they considered windows of offset size *NX* and time *2 N* as illustrated in Fig. 6. On the left is the CMP gather with offset *NX*. The blocks of data from the CMP gather was used for creating a single instance of a mini-batch (multiple sequences). The right panel is the corresponding NMO velocity pick. The velocity was estimated at the centre of the window shown in magenta color. RNN took the input *X* sequentially from X_i to X_p where *i* represents the time-step index and *p* is the number of time-steps. A fully connected (FC) layer was applied after the RNN for projecting the output to the desired dimension. It is worth mentioning that during the training, mini-batches were used to update the weights in a single iteration. The method was implemented on pre-stacked 2D data provided by Geofizyka Torun Sp Z

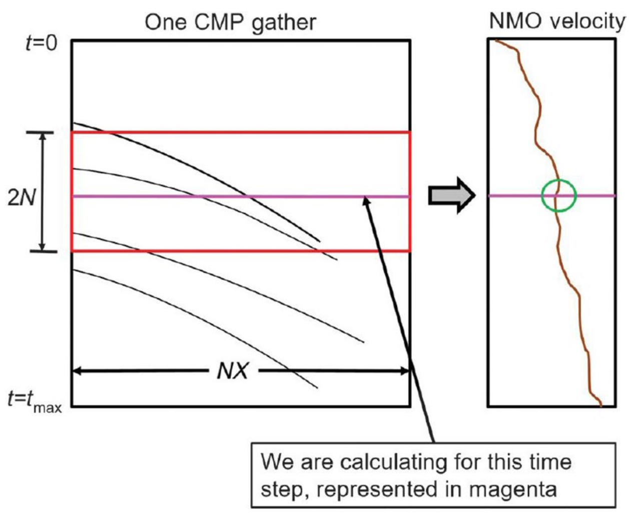
One CMP gather
NMO velocity
t=0
2N
NX
t=t_max
We are calculating for this time step, represented in magenta

Fig. 6 Illustration of the RNN input (Biswas et al. 2019b)

o.o. in Poland and available in the public domain (https://wiki.seg.org/wiki/2D_Vibroseis_Line_001). The training data were on 10% of the CMPs, which was about 80 gathers. Figure 7 shows a comparison of a hand-picked NMO with the predicted NMO by the RNN network.

In a similar fashion to Biswas et al. (2019b), Zhang et al. (2019a, 2019b) tested LSTM to automate the picking. However, they combined the LSTM with a CNN model known as YOLO (You Only Look Once) and considered the problem as an object detection problem.

More recently, Park and Sacchi (2020) utilized CNN to automatically pick the semblance. They formulated the task to be a classification problem. The network's input was defined as a pair of a guide image, *G*, which represents the velocity semblance. The target images, *T*, contained the semblance image at a specific range, τ, where τ is the zero-offset two-way travel time. This is illustrated in Fig. 8. Each image, *T*, represents the velocity in the middle of the range, τ. To reduce the computational cost, the semblance images were down sampled to 50 × 50 pixels. The output class was defined by dividing the velocity axis in the semblance panel into velocity ranges such that each range was considered as a class. They tested the method with common network structures such as LeNet-5, AlexNet, and VGG16. They concluded that VGG16 was the best choice for their implementation. The network was trained using seven synthetic quazi-horizontal models. However, with the help of transfer learning, the application was extended to more complex models.

Transfer learning is defined as follows:

Let *A* and *B* be two similar tasks. By using few samples from *B* to tune a pre-trained model on *A*, a good network that predicts from *B* can be achieved.

Park and Sacchi (2020) used six semblance panels out of 1400 from GOM data to perform transfer learning on a pre-trained network. The predicted model is similar to the NMO velocity with manual picking except at a part in the middle as shown in Fig. 9. The reason for that was explained as being due to the lack of samples on which to perform the transfer learning. This is a proof that transfer learning is vital when the model is not quazi-horizontal.

We summarize the reviewed studies on using machine learning in modeling NMO velocity in Table 1.

Velocity inversion

This section discusses some of the applications to invert for seismic velocity using ANN. These studies differ from those discussed in the previous section in that they provide a more accurate velocity that mimic the ones obtained from FWI. Some of the attempts implemented a direct inversion, which implies inputting the data and outputting the velocity

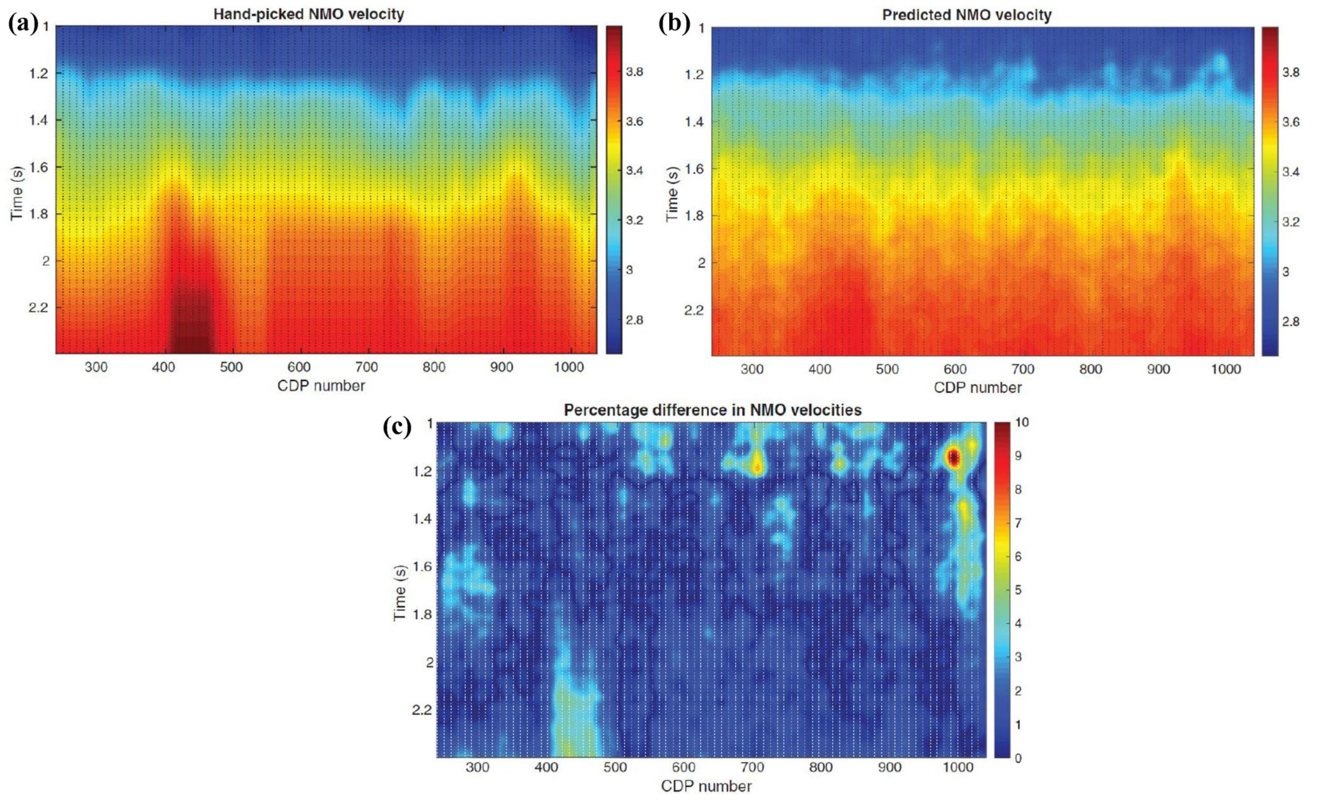

Fig. 7 **a** Hand-picked NMO velocity for Geofizyka Torun S.A, data. **b** Predicted NMO velocity by RNN. **c** is the difference between **a** and **b**. The dotted lines indicate the training area, which covers 80 CMP gather (Biswas et al. 2019b)

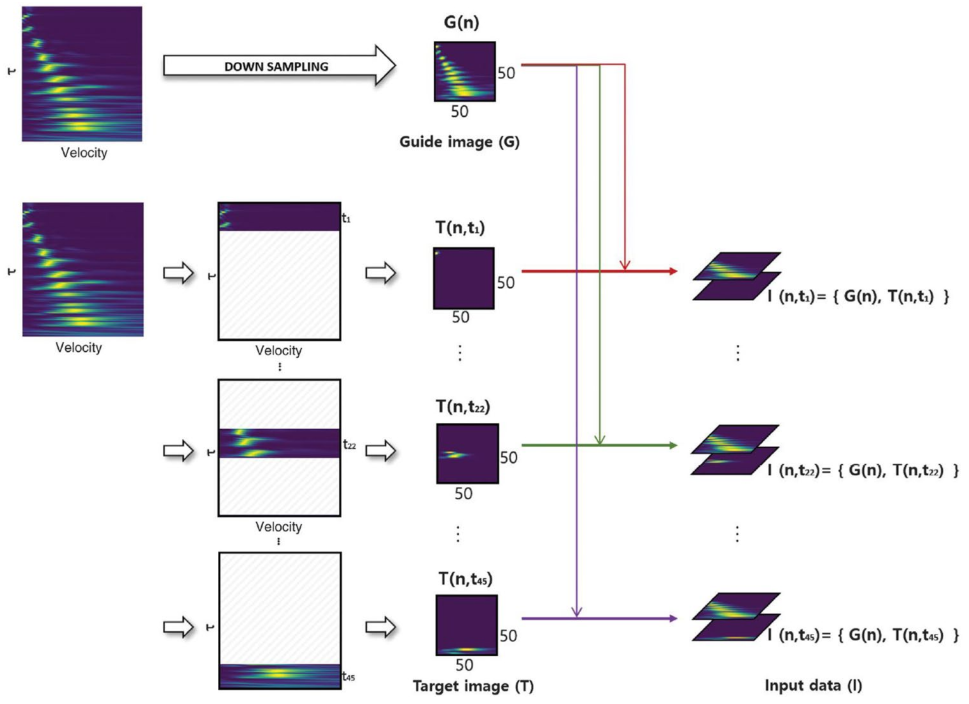

Fig. 8 Illustration of the input for the CNN network used in Park and Sacchi (2020). The guide image $G(n)$ containing the semblance is paired with the target image $T(n,t)$, which contained the semblance at a specific range τ. n is the number of the CMP gather and t, which ranges from 1 to 45, is the index of the range τ which is. $I(n,t)$ is the final input data

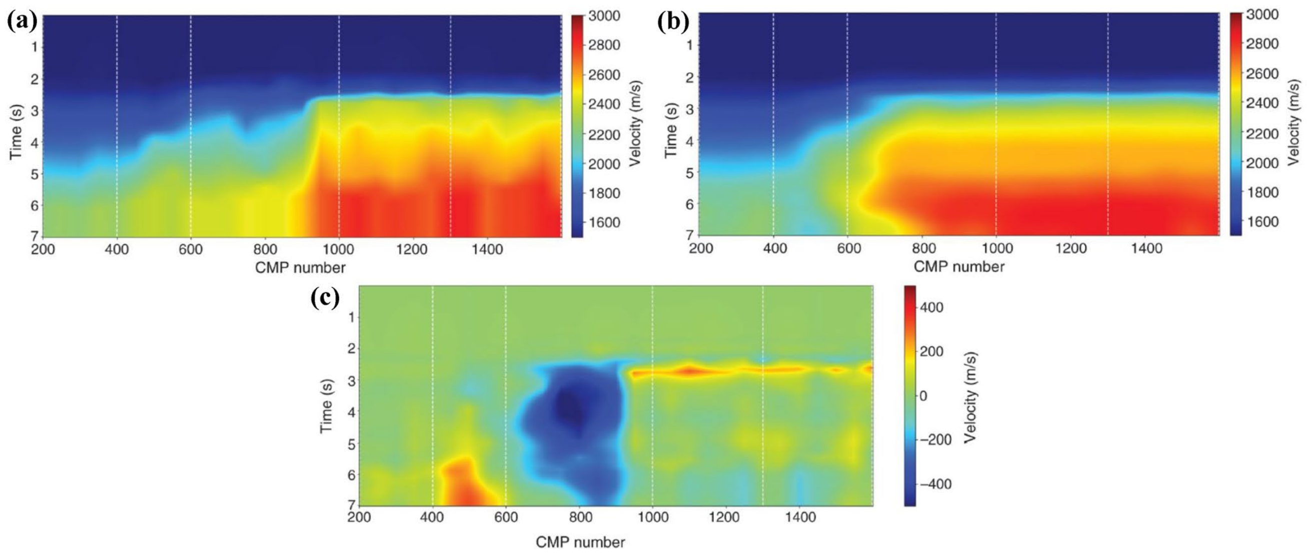

Fig. 9 **a** is the hand-picked NMO velocity model for GOM data. **b** is the predicted velocity using CNN. **c** is the difference between a and b. Transfer learning is implemented using 6 CMP gathers indicated by the dotted lines (Park and Sacchi 2020)

Table 1 Summary of the reviewed work in ML applications for modeling NMO velocity

Authors	ML Type	Proposed method	Notes
Smith (2017)	Unsupervised	*K*-means	Used different attributes besides the semblance
Wei et al. (2018)	Unsupervised	*K*-means	Used few picked semblances as a guide
Chen (2018)	Unsupervised	*K*-means	Applied a threshold to keep the high energy points
Bin Waheed et al. (2019)	Unsupervised	*K*-means + DBSCAN	Compared K-means and DBSCAN
Ma et al. (2018)	Supervised	CNN	Used CMP gathers to learn the NMO velocity
Biswas et al. (2019b)	Supervised	RNN	Used CMP gathers to learn the NMO velocity
Zhang et al. (2019a, 2019b)	Supervised	RNN + CNN	Used CMP gathers to learn the NMO velocity
Park and Sacchi (2020)	Supervised	CNN	Automated the picking from the semblance

model. Examples of this method are found in Araya-Polo et al. (2018), Yang and Ma (2019), Biswas et al. (2019b), and Sun and Alkhalifah (2020). Others utilized ANN for regularization, manipulating the gradients, extrapolating to low frequency, and adding prior based on ML. Examples of this approach are found in Jin et al. (2018), Hu et al. (2019), Lewis and Vigh (2017), Sun and Demanet (2018), Ovcharenko et al. (2019), Sun and Alkhalifah (2019a), Haber and Tenorio (2003), and Zhang and Alkhalifah (2019). The later type is referred to in this paper as ML-assisted velocity inversion. Each of these is discussed in more details.

Direct inversion

Araya-Polo et al. (2018) proposed an FC layer of ANN to predict the velocity directly from the shot gathers. They suggested extracting features from the shot gathers as doing so helped the training to converge faster and more accurate. To achieve that, they converted the data to a semblance cube and used it as features since the semblance contains patterns related to the velocity. The label for the network was the ground truth velocity. They further used three FC layers with dropout and batch normalization to test the approach. They conducted two experiments to test this method. In the first experiment, the output was a continuous-valued image and the label was composed of discrete values containing velocities. This case needed a post-processing procedure such as K-means segmentation to be applied to the output velocity images. In the second experiment, the actual labels and the predicted velocities were of continuous values. In addition, salt-bodies were included in some of the models. The two experiments performed similarly for layered velocity models. It did not perform well in the cases containing salt bodies. Salt bodies are typically challenging to invert for even in conventional approaches.

Without using any features, Wu et al. (2018) and Yang and Ma (2019) inverted for the velocity directly from the raw data. They recommended the use of CNN. While the former focused on inverting for models containing faults, the latter inverted for models containing salt bodies. Yang and Ma (2019) used a modified U-net architecture. In a typical U-net (Ronneberger et al. 2015), the input and the output are in the same dimension. In this work, the input was in space–time (x, t) while the output was in space-depth (x, z) dimension. The samples used for training are 2D synthetic velocity containing salt bodies. They used different shots generated from the same model as channels for the input. Therefore, the number of channels for each input was the same as the number of shots. The labels were the velocity models used to generate the data. The results showed promising capability of obtaining the velocity model directly from the raw data. The trained network was then used as an initial network for a different training set from SEG/EAGE salt models (Aminzadeh et al. 1996). The prediction of the network was not as good. According to Yang and Ma (2019), this may be due to the lack of training models.

In FWI, the governing equation for the problem, which is mainly the wave equation, is well known. It would be desirable to take advantage of that and use a physics-guided machine learning approach. Biswas et al. (2019a) inverted for the velocity based on the physics using an encoder-decoder CNN network. In this technique, the input dimension, which is the seismic data, was reduced in the encoder part and then restored back in the decoder part. The output model was then used in the wave equation to generate a synthetic model, and the difference between the input and the generated data was used to compute the gradient. It should be noted that this is an unsupervised approach as there was no label. Rather, the physics was used to compute the gradient and update the network.

Several researchers in the area of FWI such as Van Leeuwen and Herrmann (2013), Alkhalifah and Song (2019), and Sun and Alkhalifah (2019b) proposed using robust misfit functions to overcome the limitations of the conventional least-squared objective functions. In line with this, Sun and Alkhalifah (2020) proposed to learn a more robust objective function using the concept of meta-learning. They formulated the problem to find the optimal objective (minimum loss) by replacing the conventional L2 norm with an ANN model. They formulated the meta-loss such that the network mimicked the behavior of the optimal transport of the matching filter misfit function (Sun and Alkhalifah 2019b). As a simple test, they applied the method on a simplified FWI by inverting only for a travel time shift between two traces. They plotted the learned misfit function at the first epoch and after 250 epochs and then compared it with the L2 misfit. The ML-misfit after 250 epochs showed better convexity than the L2 objective. They then learned the objective function using random 2D horizontal layers and inferred on the Marmousi model. Since the low frequencies were missing from the data, they introduced the well-known cycle-skipping problem. Because of this, the conventional FWI result was cycle-skipped while the ML-misfit inversion was not. This suggested that the learned misfit was more robust than the L2 objective.

Velocity inversion assisted by machine learning

In the theory of FWI, the multi-scale approach (Bunks et al. 1995) suggested starting the inversion from low frequencies and progressively including the high frequencies until the whole bandwidth of the data has been used. Despite this guarantee a stable convergence, the data often lack the low frequencies. Many researchers have attempted to extrapolate the missing low frequencies from the high frequencies (Hu et al. 2019; Wu et al. 2014). The effort had been limited to the single scattering assumption, known as the Born approximation, and the acquisition. Ovcharenko et al. (2018) used a feed-forward ANN model to extrapolate the low frequencies. Thereafter, and for computational efficiency for large inputs, they suggested using CNN for the extrapolation (Ovcharenko et al. 2019). They formulated the CNN such that the input is the high-frequency contents of a shot gather. The output was a single low frequency.

The network used in this approach consisted of four convolutional blocks, followed by pooling layers, and then two FC layers. The data used for training were generated with random models. The network could only extrapolate to a single frequency, implying that there were individual CNNs for each target frequency. They tested the network on the central BP 2004 model, which contained a large salt body. The available frequency bandwidth in the seismic data ranged from 2 to 4.5 Hz. They extrapolated the frequencies to 0.25, 0.5 and 1 Hz. The error of extrapolation increased at higher frequencies. This was possibly caused by the introduction of more complex contributions of subsurface features into the total misfit (Ovcharenko et al. 2019). For FWI applications, the extrapolated low frequencies data are typically first used in inversion. Then, the final inversion result was used as input for the subsequent higher frequency until all the bandwidth was covered. They used this approach to invert for the BP velocity model. The final result successfully reconstructed the salt body, which would have been difficult using the available frequency that started from 2 Hz.

There is a lot of information that can be used to impose constraints and regularize FWI. Such constraints include geological information. The equation currently used to connect different data information is based on some surface assumptions. To better connect the data, statistical principles have been effectively used to merge different information by using deep learning. Zhang and Alkhalifah (2019) learned the probabilities of the facies for each P-velocity (V_p) and S-velocity (V_s) from a nearby well. They then mapped the probabilities into the whole estimated model by using a

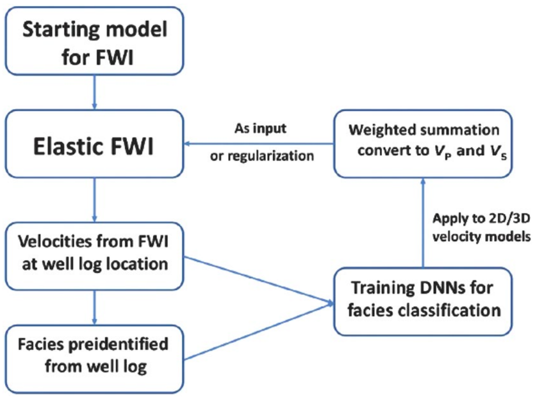
Starting model for FWI
Elastic FWI
As input or regularization
Weighted summation convert to V_P and V_S
Apply to 2D/3D velocity models
Velocities from FWI at well log location
Training DNNs for facies classification
Facies preidentified from well log

Fig. 10 Workflow to regularize FWI using facies from a well log (Zhang and Alkhalifah 2019)

weighted sum, $\sum p_i v_i$, where p is the probability of the individual facies, i. They used a network with four hidden layers, 64 neurons in each layer, and three input features, namely V_p, V_s and V_s/V_p, to output the facies. The algorithm is summarized below, and the flowchart is presented in Fig. 10:

1. Perform elastic FWI.
2. Extract facies information from a well log or any other sources.
3. Choose vertical profiles near the well from the estimated model in step 1. Then build the connection between these estimates and the interpreted facies by training a feed-forward ANN.
4. Use the trained network to predict the facies for the whole model and use a weighted summation to generate the P-velocity (V_p) and S-velocity (V_s).
5. Use the converted V_p and V_s velocities as input or regularization for another cycle of FWI.
6. Repeat the process if apparent error estimation exists.

They tested the above methodology on the BigSky field data. They manually interpreted 11 facies from a well log to obtain the initial model for the FWI by smoothening the velocity. They found that the inversion results along with facies were more accurate and higher in resolution than the conventional FWI. Figure 11 shows their inversion result. The yellow arrow points to high-resolution anomaly that was not capture by the conventional inversion.

The results of the literature survey discussed above are summarized in Table 2.

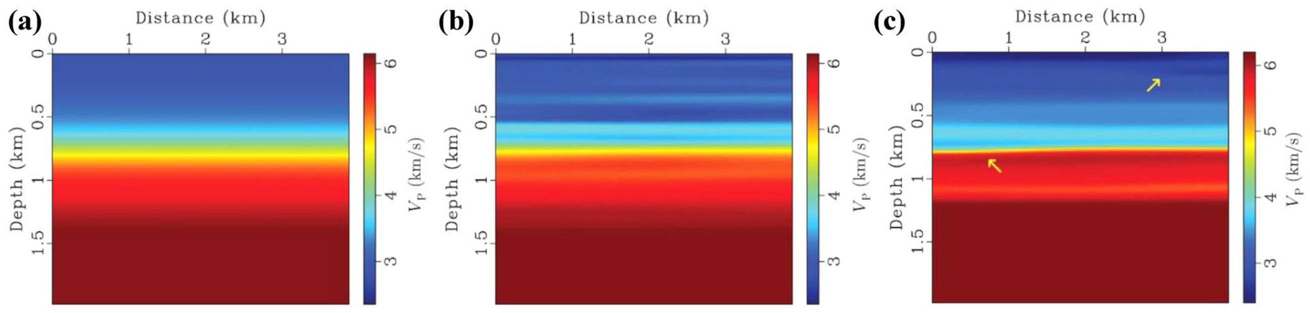

Fig. 11 V_p velocity of BigSky data for **a** initial model, **b** the estimated FWI without facies and **c** with facies. Yellow arrow indicates lateral anomaly predicted by the network (Zhang and Alkhalifah 2019)

Table 2 Summary of the reviewed ML applications to invert for the velocity

	Authors	ML type	Proposed Method	Notes
Direct Inversion	Araya-Polo et al (2018)	Supervised	Fully connected	Converted data to semblance cube
	Wu et al. (2018)	Supervised	Convolutional neural network	Used raw data to predict models with faults
	Yang and Ma (2019)	Supervised	Convolutional neural network	Used raw data to predict models with salts
	Biswas et al. (2019a)	Supervised	Convolutional neural network	Unsupervised learning by using the physics
	Sun and Alkhalifah (2020)	Semi-supervised	Meta-learning	Learned the objective function for the inversion
ML-Assisted	Ovcharenko et al. (2018)	Supervised	Fully connected	Extrapolated to low frequency
	Ovcharenko et al. (2019)	Supervised	Convolutional neural network	Extrapolated to low frequency
	Zhang and Alkhalifah (2019)	Supervised	Fully connected	Mapped a well-log facies to velocity model

Remaining gaps in the current level of machine learning applications in seismic velocity modeling

Most of the work reviewed in this paper used insufficient data to train the models. It is well known that the performance of machine learning methods usually improves with the addition of more data in terms of quantity, quality, and relevance of the features. The data used often belonged to the same velocity model, which would adversely affect the generalization of the models. Transfer learning could be a solution to use the network in different velocity models. It does not guarantee good estimation if it did not capture the new features in the new model. In most of the ML velocity applications discussed above, the training is usually performed on random synthetic models, which may not be realistic. A typical case is holding some assumption such as invariant lateral velocity like in Park and Sacchi (2020). Testing the application on real data set can be very different as the real data are contaminated with noise. The ML applications discussed above only considered simple cases where the medium is isotropic and acoustic. For example, NMO applications will surely fail to flatten the hyperbola in an anisotropic medium, as they do not account for the η parameter (Alkhalifah and Tsvankin 1995). It would be noted that in the inversion applications, a network is often trained to obtain only the acoustic model from the data.

The goal of NMO is to obtain an initial estimate of the velocity model, which is an approximation process. Hence, all the methods discussed under this gave promising results with different shades of limitations. There are apparently ample rooms for improvement. In the case of the clustering method, the small variations in the data due to noise or in the diffused deep region of the semblance might lead to inaccurate predictions as observed Chen (2018) and Bin Waheed et al. (2019). In using CNN for classifying the velocity based on the semblance (Park and Sacchi 2020), the semblance panel does not account for the surrounding semblance and the lateral continuity as expected of a good processor. The network only trained on the lateral homogeneous models. This imposes some limitations on it.

One of the major limitations in the application of machine learning in velocity inversion (FWI or traveltime tomography) is that the network is only valid for the specific dataset that was used to train it. Different models have different structures yielding different signatures in the data. For example, Yang and Ma (2019) trained a CNN model to capture the salts. When tested on models that contains salt bodies with layers of sediments, it failed to recover the layers. Besides, when the input of the network was shot gathers from raw data, the network was restricted to take a fixed number of channels (shots from the same velocity model). Including more shots from different locations would add more information and improve the generalization capability of the velocity model. However, this would be limited by the memory of the GPU. An alternative approach would be to extract features from the data. For example, Araya-Polo et al. (2018) only used one feature. From the theory of ML, extracting more features will definitely help the model to be more accurate. In addition, if the ML method used for inversion is guided by the physics rather than being completely driven by data such as in (Biswas et al. 2019a), then it would be prone to the cycle skipping problem like in the conventional FWI method.

Recommendations for future improvements

In this paper, we have presented a comprehensive review of some significant machine learning applications in velocity model building. We tracked and critically reviewed the evolution of various efforts directed toward automating the velocity picking for NMO velocity. The review exercise revealed that most efforts directed toward applying ML techniques in velocity modeling are limited to the traditional single-instance supervised learning techniques namely the families of ANN (feed-forward back-propagation, RNN, and CNN). Traditional clustering techniques, namely k-mean and DBSCAN, are also used to apply unsupervised learning. The review also revealed that some of the efforts used ML techniques either as the main engine for the inversion or to assist the traditional methods. We identified some gaps that remain to be filled as well as limitations that need to be improved on.

Based on the identified gaps and limitations, and in view of the need to accelerate the digital transformation agenda of the petroleum industry, we present the following recommendations to improve future applications of machine learning in seismic velocity modeling and estimation:

1. Since most of the reviewed applications of machine learning techniques used very limited amount of data for training, it might be useful as a guide to refer to the recommendations of Anifowose and Abdulraheem (2010) and Anifowose et al. (2017a, 2017b) on the minimum amount of data considered sufficient for shallow networks and models.
2. Going beyond using only the semblance for NMO velocity with clustering methods, the feasibility of using other features such as coherency and some measures of continuity for input could be investigated.
3. Using big data from different velocity models is a necessary requirement for achieving better generalization of the ANN and CNN models.

4. Using a 3D semblance cube instead of 2D and applying a semantic segmentation rather than classification might better account for continuity and obtain more accurate velocity.
5. For velocity inversion methods that use the raw shot gathers as input, dimensionality reduction can be used as a pre-processing step to utilize only the most significant shots for better accuracy and to minimize memory utilization. Principal component analysis, discrete cosine transform, and auto-encoders are examples of such dimensionality reduction tools. Doing this will not only minimize memory utilization but will also reduce the computational cost and increase the efficiency.
6. Since problem formulation is more of art than science, solutions to different problems are formulated in different ways. These include using different input features and target variables, different learning types (regression or classification), and different learning methods (supervised or unsupervised). This suggests that more advanced, robust, and state-of-the-art ML techniques such as SVM, random forest, light gradient boosted machine (LightGBM), extreme gradient boosted machine (XGBoost), and extreme learning machines (ELM) can be utilized with evidently more effective formulations.
7. Most of the work done so far has been with ANN. Other techniques such as those mentioned above are more robust and powerful in classification problem and can be used in place of ANN. Despite this, they have been largely underutilized in seismic velocity modeling. SVM, in particular, may not require large dataset unlike in the case of ANN (Shao and Lunetta 2012). Random forest and ELM have been presented to be robust and have the capability to avoid overfitting (Zhu et al. 2005; Bernard et al. 2012; Liu et al. 2013). LightGBM (Zhang et al. 2019a, 2019b) and XGBoost (Song et al. 2019) have proved to be robust and powerful in other fields.
8. The applications should be very explicit on how the models used for training are generated. Realistic models should contain some earth structures such as faults, anticlines, and salt bodies. Oftentimes, machine learning methodologies are validated on specific models containing one structure such as salt body. Generating more realistic models that combine more earth structures in a single model is a huge-demand and much-desired area of research that will not only benefit training the machine learning models but also prove to be useful to validate any general theory on the generated models.
9. The ML techniques applied so far in seismic velocity modeling are single-instance models. Since these techniques have their respective areas of strengths and weaknesses, we recommend the application of hybrid and ensemble learning algorithms (Anifowose et al. 2017a, 2017b). These new algorithms have the capability to combine the respective benefits of existing techniques by complementing the weaknesses of some by the strengths of the others.

Conclusions

We reviewed some machine learning applications in velocity modeling. These applications include automation of semblance picking for NMO velocity, obtaining the velocity directly from seismic data, and assist in inverting the velocity. Various machine learning technique are used in these applications, namely unsupervised, supervised, and semi-supervised methods. We also provided some recommendation that can be used for the future.

The field of artificial intelligence is evolving very fast, which raise the potential of improving the applications further. We hope that this paper will be of benefit, especially to young professionals to understand the evolution of machine learning applications in seismic velocity modeling from the inception to the current development. It will also benefit researchers in this field to direct their efforts toward providing complete automation of and more robust solutions to various seismic velocity estimation challenges.

Funding None.

Declarations

Conflict of interest The authors have no conflicts of interest to declare that are relevant to the content of this article.

References

Anifowose FA, Labadin J, Abdulraheem A (2017a) Ensemble machine learning: an untapped modeling paradigm for petroleum reservoir characterization. J Petrol Sci Eng 151:480–487

Ahmed KN, Razak TA (2016) An overview of various improvements of DBSCAN algorithm in clustering spatial databases. Int J Adv Res Comput Commun Eng 5

Alkhalifah T, Song C (2019) An efficient wavefield inversion: using a modified source function in the wave equation. Geophysics 84:R909–R922

Alkhalifah T, Tsvankin I (1995) Velocity Analysis for Transversely Isotropic Media. Geophysics 60:1550–1566

Aminzadeh F, Burkhard N, Long J, Kunz T, Duclos P (1996) Three dimensional SEG/EAGE models—an update. Leading Edge 15:131–134

Andrychowicz M, Denil M, Gomez S, Hoffman MW, Pfau D, Schaul T, Shillingford B, De Freitas N (2016) Learning to learn by gradient descent by gradient descent. Adv Neural Inform Process Syst 3981–3989

Anifowose F, Abdulraheem A (2010) How small is a "Small Data"? In: Paper # OGEP-2010–043 prepared for presentation at the 2nd Saudi meeting on oil and natural gas exploration and production technologies (OGEP 2010) held at the King Fahd University of Petroleum & Minerals (KFUPM), Dhahran, Kingdom of Saudi Arabia, December 18–20

Anifowose F, Khoukhi A, Abdulraheem A (2017b) Investigating the effect of training–testing datastratification on the performance of soft computing techniques: an experimental study. J Exp Theor Artif Intell 29:517–535. doi: https://doi.org/10.1080/0952813X.2016.1198936

Araya-Polo M, Jennings J, Adler A, Dahlke T (2018) Deep-learning tomography. Leading Edge 37:58–66

Asnaashari A, Brossier R, Garambois S, Audebert F, Thore P, Virieux J (2013) Regularized seismic full waveform inversion with prior model information. Geophysics 78:R25–R36

Bernard S, Adam S, Heutte L (2012) Dynamic random forests. Patt Recogn Lett 33:1580–1586

bin Waheed U, Al-Zahrani S, Hanafy SM, 2019, Machine learning algorithms for automatic velocity picking: K-means vs. dbscan, in SEG technical program expanded abstracts 2019. Soc Explor Geophys 5110–5114

Bishop CM (2006) Pattern recognition and machine learning. Springer

Biswas R, Sen MK, Das V, Mukerji T (2019a) Prestack and poststack inversion using a physics-guided convolutional neural network. Interpretation 7:SE161–SE174

Biswas R, Vassiliou A, Stromberg R, Sen MK (2019b) Estimating normal moveout velocity using the recurrent neural network. Interpretation 7:T819–T827

Bradley PS, Fayyad U, Reina C et al (1998) Scaling EM (expectation-maximization) clustering to large databases

Bunks C, Saleck FM, Zaleski S, Chavent G (1995) Multiscale seismic waveform inversion. Geophysics 60:1457–1473

Chen Y (2018) Automatic semblance picking by a bottom-up clustering method: SEG 2018 Workshop: SEG maximizing asset value through artificial intelligence and machine learning, Beijing, China, 17–19 September 2018. Society of exploration geophysicists and the Chinese geophysical society, pp 44–48

De Oliveira JV, Pedrycz W (2007) Advances in fuzzy clustering and its applications. John Wiley & Sons

Fish BC, Kusuma T (1994) A neural network approach to automate velocity picking, in SEG Technical Program Expanded Abstracts 1994. Soc Explor Geophys 185–188

Gan G, Ma C, Wu J (2007) Data clustering: theory, algorithms, and applications. Siam 20

Haber E, Tenorio L (2003) Learning regularization functionals—a supervised training approach. Inv Problems 19:611

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9:1735–1780

Hole J (1992) Nonlinear high-resolution three-dimensional seismic travel time tomography. J Geophys Res Solid Earth 97:6553–6562

Hu W, Jin Y, Wu X, Chen J (2019) A progressive deep transfer learning approach to cycle-skipping mitigation in fwi, in SEG technical program expanded abstracts 2019. Soc Explor Geophys 2348–2352

Jain AK (2010) Data clustering: 50 years beyond k-means. Patt Recogn Lett 31:651–666

Jin Y, Hu W, Wu X, Chen J (2018) Learn low wavenumber information in fwi via deep inception based convolutional networks, in SEG technical program expanded abstracts 2018. Soc Explor Geophys 2091–2095

Kalita M, Kazei V, Choi Y, Alkhalifah T (2019) Regularized full-waveform inversion with automated salt flooding. Geophysics 84:R569–R582

LeCun Y, Bengio Y et al (1995) Convolutional networks for images, speech, and time series. Handbook Brain Theory Neural Netw 3361

Lewis W, Vigh D (2017) Deep learning prior models from seismic images for full-waveform inversion, in SEG technical program expanded abstracts 2017. Soc Explor Geophys 1512–1517

Liu M, Wang M, Wang J, Li D (2013) Comparison of random forest, support vector machine and back propagation neural network for electronic tongue data classification: application to the recognition of orange beverage and Chinese vinegar. Sensors Actuat B Chem 177:970–980

Ma Y, Ji X, Fei TW, Luo Y (2018) Automatic velocity picking with convolutional neural networks, in SEG technical program expanded abstracts 2018. Soc Explor Geophys 2066–2070

Maclaurin D, Duvenaud D, Adams R (2015) Gradient-based hyperparameter optimization through reversible learning. Int Conf Mach Learn 2113–2122

Maniar H, Ryali S, Kulkarni MS, Abubakar A (2018) Machine-learning methods in geoscience, in SEG technical program expanded abstracts 2018. Soc Explor Geophys 4638–4642

Ovcharenko O, Kazei V, Peter D, Zhang X, Alkhalifah T (2018) Low-frequency data extrapolation using a feed-forward ann. In: 80th EAGE conference and exhibition 2018, European association of geoscientists & engineers, pp 1–5

Ovcharenko O, Kazei V, Kalita M, Peter D, Alkhalifah T (2019) Deep learning for low-frequency extrapolation from multioffset seismic data. Geophysics 84:R989–R1001

Park MJ, Sacchi MD (2020) Automatic velocity analysis using convolutional neural network and transfer learning. Geophysics 85:V33–V43

Qi J, Zhang B, Lyu B, Marfurt K (2020) Seismic attribute selection for machine-learning-based facies analysis. Geophysics 85:O17–O35

Rokach L, Maimon O (2005) Clustering methods. In: Data mining and knowledge discovery handbook. Springer, pp 321–352

Ronneberger O, Fischer P, Brox T (2015) U-net: convolutional networks for biomedical image segmentation. In: International conference on medical image computing and computer-assisted intervention. Springer, pp 234–241

Salvador S, Chan P (2004) Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms. In: 16th IEEE international conference on tools with artificial intelligence. IEEE, pp 576–584

Sava P, Biondi B (2004) Wave-equation migration velocity analysis. I. Theory. Geophys Prospect 52:593–606

Schmidt J, Hadsell FA (1992) Neural network stacking velocity picking, in SEG technical program expanded abstracts 1992. Soci Explor Geophys 18–21

Shao Y, Lunetta RS (2012) Comparison of support vector machine, neural network, and cart algorithms for the land-cover classification using limited training data points. ISPRS J Photogramm Remote Sensing 70:78–87

Smith K (2017) Machine learning assisted velocity autopicking, in SEG technical program expanded abstracts 2017. Soc Explor Geophys 5686–5690

Song Y, Jiao X, Qiao Y, Liu X, Qiang Y, Liu Z, Zhang L (2019) Prediction of double-high biochemical indicators based on LightGBM and XGBoost. In: Proceedings of the 2019 international conference on artificial intelligence and computer science, July 2019, pp 189–193

Sun B, Alkhalifah T (2019a) Ml-descent: an optimization algorithm for FWI using machine learning, in SEG technical program expanded abstracts 2019. Soc Explor Geophys 2288– 2292

Sun B, Alkhalifah T (2019b) Robust full-waveform inversion with radon-domain matching filter. Geophysics 84:R707–R724

Sun B, Alkhalifah T (2020) Ml-misfit: learning a robust misfit function for full-waveform inversion using machine learning. In: 82nd EAGE Annual Conference & Exhibition (Vol. 2020, No. 1, pp. 1–5). European Association of Geoscientists & Engineers

Sun H, Demanet L (2018) Low frequency extrapolation with deep learning, in SEG technical program expanded abstracts 2018. Soc Explor Geophys 2011–2015

Symes WW (2008) Migration Velocity Analysis and Waveform Inversion. Geophys Prospect 56:765–790

Tarantola A (1984) Inversion of seismic reflection data in the acoustic approximation. Geophysics 49:1259–1266

Van Leeuwen T, Herrmann FJ (2013) Mitigating local minima in full-waveform inversion by expanding the search space. Geophys J Int 195:661–667

Wei S, Yonglin O, Qingcai Z, Jiaqiang H, Yaying S (2018) Unsupervised machine learning: K-means clustering velocity semblance auto-picking. In: 80th EAGE conference and exhibition 2018, European association of geoscientists & Engineers, pp 1–5

Wrona T, Pan I, Gawthorpe RL, Fossen H (2018) Seismic facies analysis using machine learning. Geophysics 83:O83–O95

Wu R-S, Luo J, Wu B (2014) Seismic envelope inversion and modulation signal model. Geophysics 79:WA13–WA24

Wu Y, Lin Y, Zhou Z (2018) Inversionnet: accurate and efficient seismic waveform inversion with convolutional neural networks, in SEG technical program expanded abstracts 2018. Soc Explor Geophys 2096–2100

Wu X, Liang L, Shi Y, Fomel S (2019) Faultseg3d: Using synthetic data sets to train an end-to-end convolutional neural network for 3D seismic fault segmentation. Geophysics 84:IM35–IM45

Xiong W, Ji X, Ma Y, Wang Y, AlBinHassan NM, Ali MN, Luo Y (2018) Seismic fault detection with convolutional neural network. Geophysics 83:O97–O103

Xuan X, Murphy K (2007) Modeling changing dependency structure in multivariate time series. In: Proceedings of the 24th international conference on Machine learning, pp 1055–1062

Yang F, Ma J (2019) Deep-learning inversion: a next-generation seismic velocity model building method. Geophysics 84:R583–R599

Yilmaz O (1987) Seismic data processing. Investig Geophys

Zeng Y, Jiang K, Chen J (2019) Automatic seismic salt interpretation with deep convolutional neural networks. In: Proceedings of the 2019 3rd international conference on information system and data mining, pp 16–20

Zhang Z-D, Alkhalifah T (2019) Regularized elastic full-waveform inversion using deep learning. Geophysics 84:R741–R751

Zhang J, Mucs D, Norinder U, Svensson F (2019a) LightGBM: an effective and scalable algorithm for prediction of chemical toxicity-application to the Tox21 and mutagenicity data sets. J Chem Inf Model 59(10):4150–4158

Zhang H, Zhu P, Gu Y, Li X (2019) Automatic velocity picking based on deep learning, in SEG technical program expanded abstracts 2019. Soc Explor Geophys 2604–2608

Zhu Q-Y, Qin AK, Suganthan PN, Huang G-B (2005) Evolutionary extreme learning machine. Pattern Recogn 38:1759–1763