Seismic Velocity Modelling in the Era of Digital Transformation: The Role of Machine Learning and Recommendations for Full Attainment

Abdullah Alali[1] and Fatai Anifowose[2]

1. *King Abdullah University of Science and Technology (KAUST).*
2. *Saudi Aramco.*

Abstract

Due to recent advances in computational resources and big data technology, machine learning (ML) algorithms have been widely embraced in the geoscience community. Several applications, including seismic processing and interpretation, have utilized several ML techniques. This paper focuses on one of the most crucial processing steps in seismic data processing, namely velocity modelling. Often, the first velocity model obtained from seismic data is the normal moveout (NMO) velocity. It is used to flatten the hyperbolic events in the data. Despite that NMO velocity is not accurate in complex media, it is often used as an initial guess for further velocity modelling methods such as traveltime tomography and full-waveform inversion (FWI). All the velocity modelling methods have certain limitations, which include extensive human intervention in form of picking semblances in NMO velocity analysis or converging to a local minimum as in FWI. In this paper, we critically review the studies that utilized the ML techniques to overcome the velocity modelling limitations. We also discuss ML applications that utilize an inversion-like velocity such as those obtained from tomography and FWI. The reviewed applications cover different types of learning: supervised, unsupervised, and semi-supervised. In view of contributing to the digital transformation agenda of the petroleum industry, this paper concludes with a set of recommendations to overcome the prevailing challenges and for the implementation of more advanced ML technologies. We hope that the recommendations will help to achieve complete automation of the NMO velocity and further enhance the performance of ML applications as used in the FWI framework.

Keywords: Machine learning, Digital transformation, Velocity analysis, NMO velocity, Velocity inversion, Full-waveform inversion.

1.0 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 lateral homogeneity and fails to estimate complex structures. However, it provides a smooth velocity due to the interpolation between the CMP gathers.

Traveltime tomography estimates the velocity model by inverting the traveltime using the Eikonal equation and ray-tracing methods (Hole, 1992). These methods are non-linear and ill-posed. They are also based on the high-frequency assumption, which states that the wavelength of the seismic wave is much smaller than the minimum scale length of heterogeneity. Therefore, they fail to estimate the velocity accurately in complex regions. Other methods of velocity analysis rely on the wave equation as a carrier of information to iteratively update the velocity model (Sava and Biondi; 2004; Symes, 2008). These methods for building velocity models often require extending the domain of imaging, which is very costly for 2D and impractical for 3D volumes. Full-waveform inversion (Tarantola, 1984) provides a high-resolution velocity model by minimizing the least square misfit between observed data from the field and the modelled version. 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 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. As a pattern recognition tool, Wrona et al. (2018) used an artificial neural network (ANN) for seismic facies analysis and Qi et al. (2020) for seismic attributes selection. In petrophysical well log correlations, formation tops have been identified using unsupervised (Xuan and Murphy, 2007) or supervised (Maniar et al., 2018) learning techniques. Convolutional neural networks (CNN) was used in fault detection (Xiong et al., 2018; Wu et al., 2019), salt interpretation (Zeng et al., 2019), and many other applications. Moreover, many 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 pre-stack 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 such as clustering and supervised learning by deep neural networks. Then, we discuss the ML applications in estimating NMO velocity. After that, we review the ML applications in the more advanced velocity modelling methods: traveltime tomography and FWI. Finally, we recommend potential future applications in view of contributing to the digital transformation agenda of the petroleum industry.

2.0 Overview of ML Techniques Commonly 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 to estimate the "loss" between the model prediction, T', and the actual target, *T*, the model is optimally tuned to predict the desired output. The goal of the tuning process is to keep the loss within a certain threshold or as low as practically possible. 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 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 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 the task of learning an action in a certain situation to maximize the rewards. Here, there is no optimal target, *T*, but the machine must learn them by a process of trial and error.

A recent ML technique known as meta-learning or "learning to learn" is 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 (e.g., loss, learning rate, regularization parameter, and activation functions). Depending on how to formulate the problem, one can choose whichever hyperparameters one is interested in to optimize. A loss, referred to as meta-loss, measures the performance of the network or model in terms of how close the model prediction is to the actual target values. The algorithm starts by running series of training instances using some ML models. The meta-loss will then measure how well the ML model has succeeded in predicting the target. It would then propagate the error to update the network parameters. As one can imagine, this type of learning requires more computation as the training process is being optimized. 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 details some of the techniques used by geoscientists in velocity modelling, specifically clustering methods and ANN.

2.1 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). For a given data set, *X*, with *N* observations, the goal is to partition the data set into *K* clusters. Researchers have suggested different methods to choose the optimal number of clusters (e.g., Maclaurin et al., 2015; Salvador and Chan, 2004). However, the user often determines it by trial and error. Various clustering algorithms such as K-means, expectation maximization, density-based spatial clustering (DBCAN), 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, as we will feature them later in the velocity applications section.

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 step 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 process with three clusters.

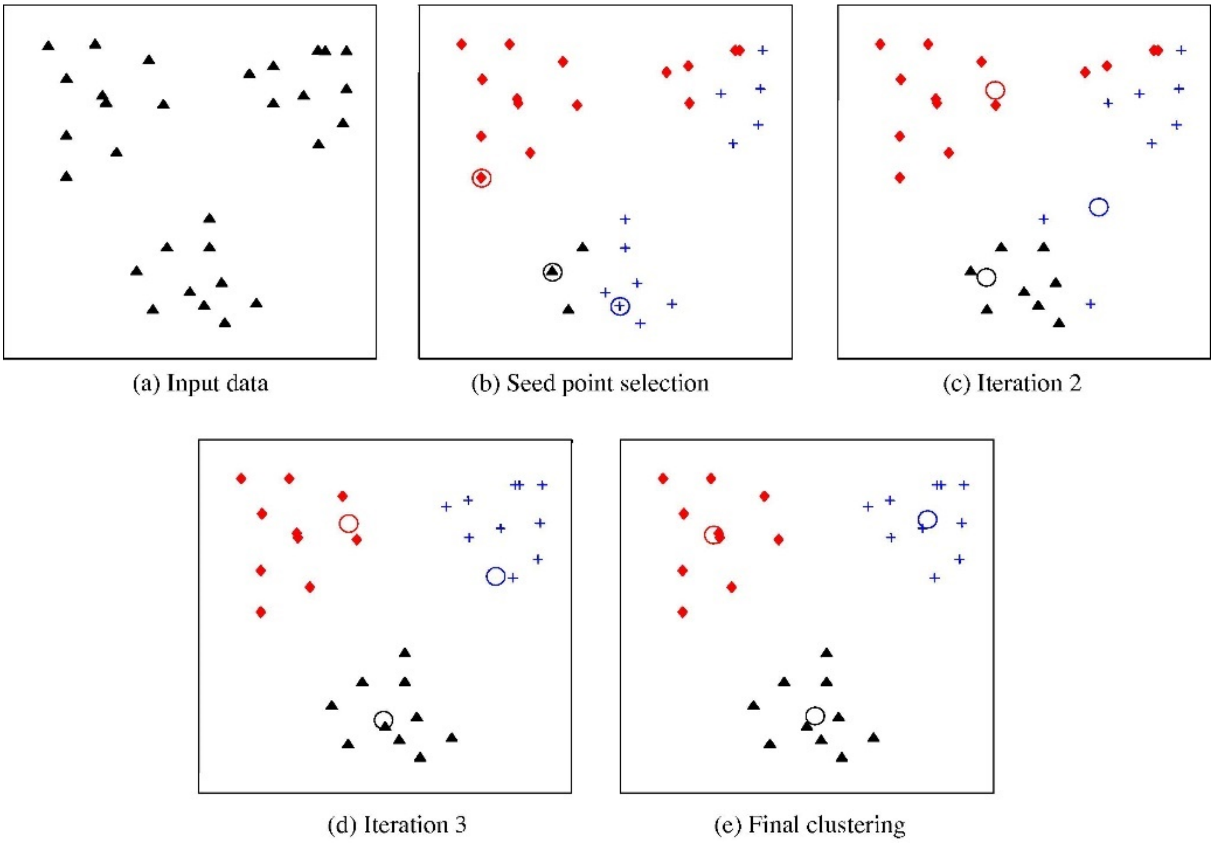

Figure 1: The K-means clustering process with $K = 3$ (Jain, 2010).

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 are more than *minPts* within the radius, ε,

2. Border points: the points that are less than *minPts* within the radius, ε,

3. Noise (outlier) points: the points that belong to neither core points nor border points.

The workflow for the method is shown in Figure 2 and it is composed of the following steps:

1. Select random points that are not assigned to a cluster or noise.
2. Compute the neighbourhood 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 become a core point.
    b. If the number of points within ε is less than minPts, 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 neighbourhood point.
5. The process is repeated until all data are assigned to a cluster or identify as outlier.

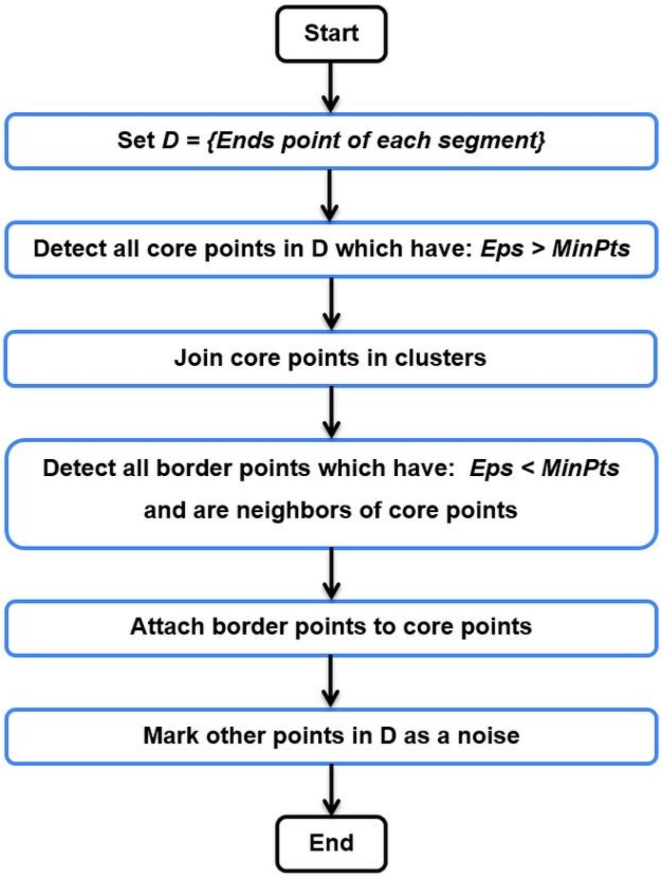

Figure 2: Flowchart for the DBSCAN clustering algorithm (El Bahi and Zatni, 2018). *Eps* is the number of points within the radius ε.

2.2 Artificial Neural Network

ANN is a powerful learning tool to approximate non-linear functions. It is considered as a supervised learning as it requires labelling the data for the training phase (Bishop, 2006). The ANN algorithm is composed of three layers: an input layer that receives the input from the user; one or more hidden layers that perform all the computations, and an output layer that produces the final results. Each layer 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 3 shows a typical ANN structure with one 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, 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, translation, and time series forecasting. RNN suffers from vanishing or exploding gradient problem. A 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).

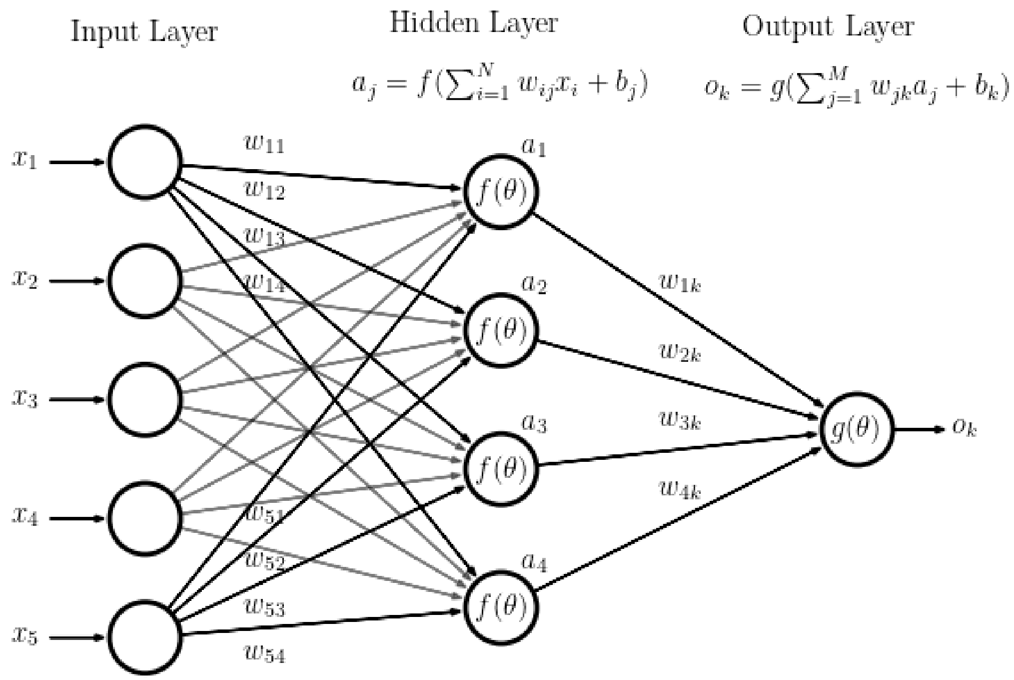

Figure 3: ANN structures with the mathematical operations inside the neuron. x_i is the input and w_{ij} are the weights. Each circle represents a single neuron.

3.0 ML Techniques Applied to Velocity Estimation

This section discusses two major types of velocity models: NMO and that based on optimization. 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 initial estimate for the subsurface model. The velocity obtained by solving an optimization problem includes traveltime tomography or full waveform inversion (FWI). This provides more accurate velocity and higher resolution model, in the case of FWI.

3.1 ML Applications in NMO Velocity Analysis

Traditionally, NMO velocity is found 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 traveltime in the other. The highest energy semblance, which indicates the velocity at a particular time, is manually picked. With the help of ML, 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 examples of the pioneers who used ANN for velocity analysis. The networks at that time were shallow and only able to extract some local information of the velocity semblance.

Many researchers suggested the use of K-means clustering as an auto-picking method for velocity analysis. Since seismic data is 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. For DBSCAN, the values of the picks for the tested radii are similar. They made a comparison between the picks for K-means with $K = 5$ and for DBSCAN with $r = 0.02$ and the true velocity.

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 was applied to the Marmousi model using ten CMP gathers for training out of 576. For each CMP gather, the moveout correction was computed using 21 trial velocities equally spaced between a fraction of 0.9 and 1.1 of the true velocity. The result produced by the estimated model was very similar to the true stacking velocity model.

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 neighbouring temporal and spatial information. Therefore, they considered windows of offset size NX and time $2N$ as illustrated in Figure 4. 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 colour. 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 o.o. in Poland and available in the public domain. The training data was 10% of the CMPs, which was about 80 gathers.

In a similar fashion to Biswas et al. (2019b), Zhang et al. (2019) 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.

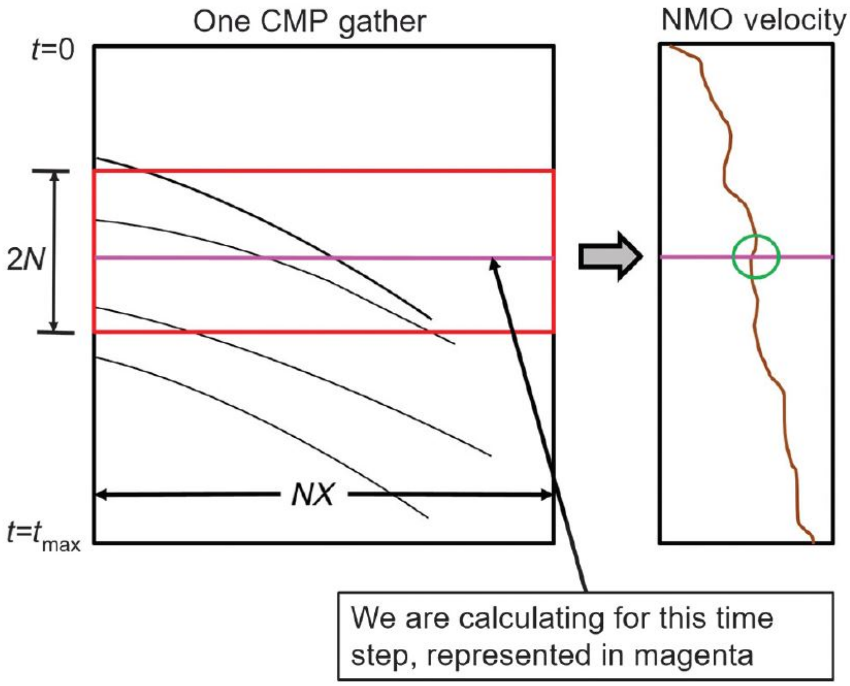

Figure 4: Illustration of the RNN input (Biswas et al., 2019b)

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 Figure 5. 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 x 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.

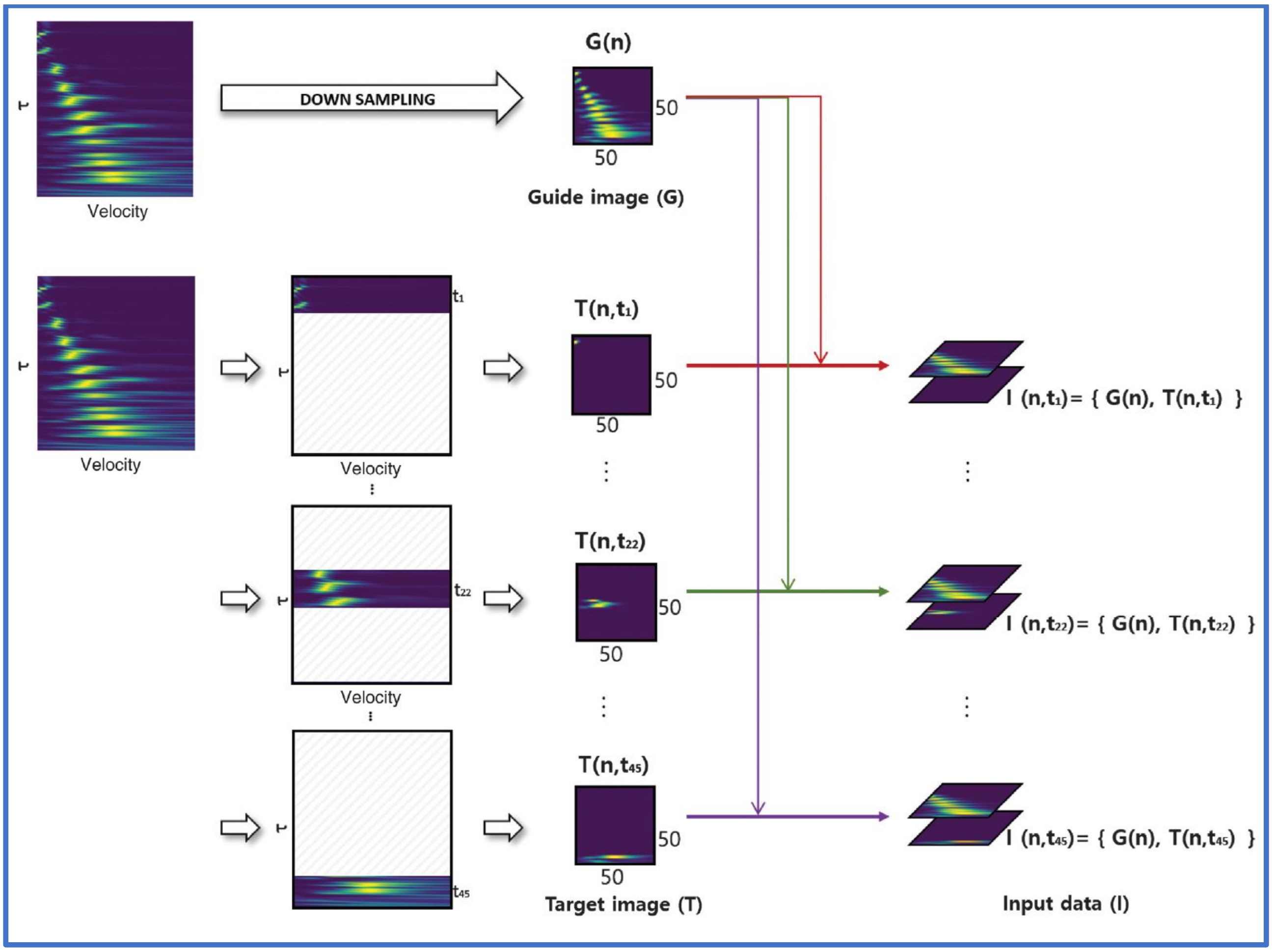

Figure 5: Illustration of the input for the CNN network used in Park and Sacchi (2020).

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. The reason for that was explained as being due to the lack of samples to perform the transfer learning on. This is a proof that transfer learning is vital when the model is not quazi-horizontal.

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

Table1: Summary of the reviewed work in ML applications for modelling NMO velocity

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

3.2 ML Application on 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 traveltime tomography or FWI. Some of the attempts implemented a direct inversion, which implies inputting the data and outputting the velocity 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.

3.2.1 Direct Inversion

Araya-Polo et al. (2018) proposed an FC layer of ANN to predict a tomography-like 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. However, 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 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 domain. However, in this work, the input was in space-time (x, t) while the output was in space-depth (x, z) domains. 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 behaviour 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.

3.2.2 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 lacks 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). However, 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 shape of the high frequencies input was "Frequencies × Receivers × 2", where "2" represents the real and the imaginary part. The output was a single low frequency with "1 × Receivers × 2" shape.

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 was 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 are a lot of information that can be used to impose constraints and regularize FWI. Such constraints include the geological information. However, the equation currently used to connect different data information are 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 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 Figure 6:

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 of the well. The inversion results with facies were more accurate and higher in resolution than the conventional FWI.

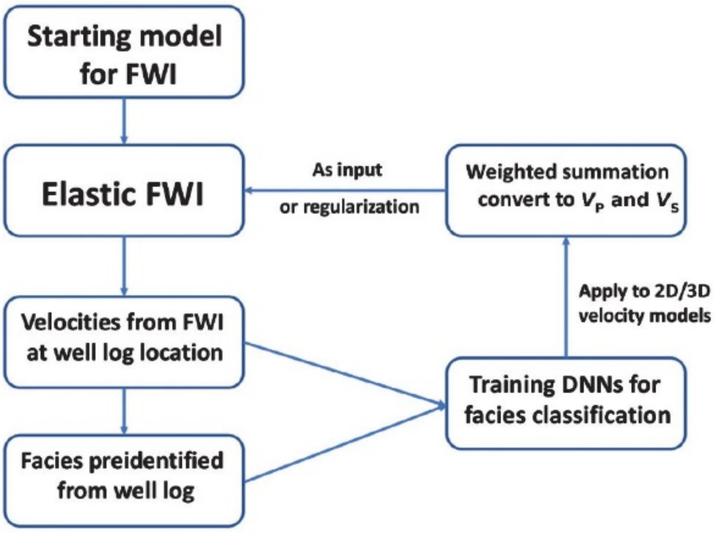

Figure 6: The workflow to regularize FWI using facies from a well log (Zhang and Alkhalifah, 2019).

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

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

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

4.0 Remaining Gaps Arising From the Current Level of Reviewed ML Applications

The goal of NMO is to obtain an initial estimate of the velocity model, which is an approximation. Hence, all the methods discussed under this gave promising results. However, there are still rooms for improvements. 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 (Chen, 2018; 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.

Most of the work reviewed in this paper used insufficient data to train the models. It is well known that the performance of ML methods usually improves with the addition of more useful data in terms of quantity and relevant 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. However, it does not guarantee good estimation if it did not capture the new features in the new model.

The major limitation in the application of ML in velocity inversion (FWI or traveltime tomography) is that the network is only valid for the specific dataset that it was trained on. 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. However, 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.

In most of the ML velocity applications discussed above, the training is usually performed in 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 is 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.

5.0 Recommendations for More Robust Future Applications

Based on the insights derived from the reviewed literature and the identified gaps, and in view of the need to contribute to the digital transformation agenda of the petroleum industry, we present a number of recommendations to improve future applications of ML in seismic velocity modelling and estimation.

1. Since most of the reviewed applications of ML 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. (2017) 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, other features such as coherency and some measures of continuity could be considered as input features.
3. Using big data from different velocity models is a necessary requirement for achieving better generalization of the ANN model.
4. Using a 3D semblance cube instead of 2D and applying a semantic segmentation rather than classification might account for continuity and obtain better 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, many approaches formulated their problems in different ways. These include using different input features and output 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, and extreme learning machines (ELM) can be utilized with formulations that are more effective.
7. Almost all the work done so far has been with ANN. Other techniques such as those mentioned above are also very powerful in classification problem and can be used similar to the way ANN was used. Despite this, they have been largely underutilized in seismic velocity modelling. 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).
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, ML methodologies are tested to 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 ML models but also is useful to test any general theory on the generated models.
9. The ML techniques applied so far in seismic velocity modelling 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., 2017). These new algorithms have the capability to combine the respective benefits of existing techniques by complementing the weaknesses in one by the strengths of the others.

6.0 Conclusion

In this paper, we have presented a comprehensive review of some significant ML applications in velocity model building. We tracked and critically reviewed the evolution of various efforts directed towards automating the velocity picking for NMO velocity. The outcome of the review revealed that most efforts directed towards applying ML techniques in velocity modelling 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 several gaps remaining to be filled as well as limitations that need to be improved upon. Based on the identified gaps and limitations, we recommended various recipes ranging from extracting more features, through using more data for training, to utilizing the latest state-of-the-art techniques based on hybrid and ensemble learning methodologies.

We hope that this paper will be of benefit especially to young professionals to understand the evolution of ML applications in seismic velocity modelling from the inception to the current development. It will also benefit researchers in this field to direct their efforts towards providing complete automation of and more robust solutions to this seismic velocity estimation challenge.

References

Ahmed, K. N., and T. A. Razak, 2016, An overview of various improvements of DBSCAN algorithm in clustering spatial databases: International Journal of Advanced Research in Computer and Communication Engineering, 5.

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

Alkhalifah, T., and I. Tsvankin, 1995, Velocity analysis for transversely isotropic media: Geophysics, 60, 1550–1566.

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

Andrychowicz, M., M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. De Freitas, 2016, Learning to learn by gradient descent by gradient descent: Advances in neural information processing systems, 3981–3989.

Anifowose, F. A., J. Labadin, and A. Abdulraheem, 2017, Ensemble machine learning: An untapped modeling paradigm for petroleum reservoir characterization: Journal of Petroleum Science and Engineering, 151, 480–487.

Anifowose, F. and A Abdulraheem, 2010, How Small is a "Small Data"?, 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., A. Khoukhi, and A. Abdulraheem, 2017a, Investigating the effect of training–testing datastratification on the performance of soft computing techniques: an experimental study: Journal of Experimental & Theoretical Artificial Intelligence,29, 517–535, DOI: 10.1080/0952813X.2016.1198936.

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

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

Bernard, S., S. Adam, and L. Heutte, 2012, Dynamic random forests: Pattern Recognition Letters,33,1580–1586.

bin Waheed, U., S. Al-Zahrani, and S. M. Hanafy, 2019, Machine learning algorithms for automatic velocity picking: K-means vs. dbscan, in SEG Technical Program Expanded Abstracts 2019: Society of Exploration Geophysicists, 5110–5114.

Bishop, C. M., 2006, Pattern recognition and machine learning: springer.

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

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

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

Bunks, C., F. M. Saleck, S. Zaleski, and G. Chavent, 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, 44–48.

De Oliveira, J. V., and W. Pedrycz, 2007, Advances in fuzzy clustering and its applications: John Wiley & Sons.

El Bahi, H., and A. Zatni, 2018, Document text detection in video frames acquired by a smartphone based on line segment detector and DBSCAN clustering: Journal of Engineering Science and Technology, 13, 540–557.

Fish, B. C., and T. Kusuma, 1994, A neural network approach to automate velocity picking, in SEG Technical Program Expanded Abstracts 1994: Society of Exploration Geophysicists, 185–188.

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

Geofizyka Torun Sp. Z O.O, Poland, data from https://wiki.seg.org/wiki/2D_Vibroseis_Line_001.

Haber, E., and L. Tenorio, 2003, Learning regularization functionals—a supervised training approach: Inverse Problems, 19, 611.

Hochreiter, S., and J. Schmidhuber, 1997, Long short-term memory: Neural computation, 9, 1735–1780.

Hole, J., 1992, Nonlinear high-resolution three-dimensional seismic travel time tomography: Journal of Geophysical Research: Solid Earth, 97, 6553–6562.

Hu, W., Y. Jin, X. Wu, and J. Chen, 2019, A progressive deep transfer learning approach to cycle-skipping mitigation in fwi, in SEG Technical Program Expanded Abstracts 2019: Society of Exploration Geophysicists, 2348–2352.

Jain, A. K., 2010, Data clustering: 50 years beyond k-means: Pattern recognition letters, 31, 651–666.

Jin, Y., W. Hu, X. Wu, and J. Chen, 2018, Learn low wavenumber information in fwi via deep inception based convolutional networks, in SEG Technical Program Expanded Abstracts 2018: Society of Exploration Geophysicists, 2091–2095.

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

LeCun, Y., Y. Bengio, et al., 1995, Convolutional networks for images, speech, and time series: The handbook of brain theory and neural networks, 3361, 1995.

Lewis, W., and D. Vigh, 2017, Deep learning prior models from seismic images for full-waveform inversion, in SEG Technical Program Expanded Abstracts 2017: Society of Exploration Geophysicists, 1512–1517.

Liu, M., M. Wang, J. Wang, and D. Li, 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 and Actuators B: Chemical,177, 970–980.

Ma, Y., X. Ji, T. W. Fei, and Y. Luo, 2018, Automatic velocity picking with convolutional neural networks, in SEG Technical Program Expanded Abstracts 2018: Society of Exploration Geophysicists, 2066–2070.

Maclaurin, D., D. Duvenaud, and R. Adams, 2015, Gradient-based hyperparameter optimization through reversible learning: International Conference on Machine Learning, 2113–2122.

Maniar, H., S. Ryali, M. S. Kulkarni, and A. Abubakar, 2018, Machine-learning methods in geoscience, in SEG Technical Program Expanded Abstracts 2018: Society of Exploration Geophysicists, 4638–4642.

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

Ovcharenko, O., V. Kazei, D. Peter, X. Zhang, and T. Alkhalifah, 2018, Low-frequency data extrapolation using a feed-forward ann: 80th EAGE Conference and Exhibition 2018, European Association of Geoscientists & Engineers, 1–5.

Park, M. J., and M. D. Sacchi, 2020, Automatic velocity analysis using convolutional neural network and transfer learning: Geophysics, 85, V33–V43.

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

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

Ronneberger, O., P. Fischer, and T. Brox, 2015, U-net: Convolutional networks for biomedical image segmentation: International Conference on Medical image computing and computer-assisted intervention, Springer, 234–241.

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

Sava, P., and B. Biondi, 2004, Wave-equation migration velocity analysis. I. theory: Geophysical Prospecting, 52, 593–606.

Schmidt, J., and F. A. Hadsell, 1992, Neural network stacking velocity picking, in SEG Technical Program Expanded Abstracts 1992: Society of Exploration Geophysicists, 18–21.

Shao, Y., and R. S. Lunetta, 2012, Comparison of support vector machine, neural network, and cart algorithms for the land-cover classification using limited training data points: ISPRS Journal of Photogrammetry and Remote Sensing,70, 78–87

Smith, K., 2017, Machine learning assisted velocity autopicking, in SEG Technical Program Expanded Abstracts 2017: Society of Exploration Geophysicists, 5686–5690.

Sun, B., and T. Alkhalifah, 2019a, Ml-descent: an optimization algorithm for FWI using machine learning, in SEG Technical Program Expanded Abstracts 2019: Society of Exploration Geophysicists, 2288– 2292.

———, 2019b, Robust full-waveform inversion with radon-domain matching filter: Geophysics, 84, R707–R724.

———, 2020, Ml-misfit: Learning a robust misfit function for full-waveform inversion using machine learning: ArXiv preprint arXiv:2002.03163.

Sun, H., and L. Demanet, 2018, Low frequency extrapolation with deep learning, in SEG Technical Program Expanded Abstracts 2018: Society of Exploration Geophysicists, 2011–2015.

Symes, W. W., 2008, Migration velocity analysis and waveform inversion: Geophysical prospecting, 56, 765–790.

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

Van Leeuwen, T., and F. J. Herrmann, 2013, Mitigating local minima in full-waveform inversion by expanding the search space: Geophysical Journal International, 195, 661–667.

Wei, S., O. Yonglin, Z. Qingcai, H. Jiaqiang, and S. Yaying, 2018, Unsupervised machine learning: K-means clustering velocity semblance auto-picking: 80th EAGE Conference and Exhibition 2018, European Association of Geoscientists & Engineers, 1–5.

Wrona, T., I. Pan, R. L. Gawthorpe, and H. Fossen, 2018, Seismic facies analysis using machine learning: Geophysics, 83, O83–O95.

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

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

Wu, Y., Y. Lin, and Z. Zhou, 2018, Inversionnet: Accurate and efficient seismic waveform inversion with convolutional neural networks, in SEG Technical Program Expanded Abstracts 2018: Society of Exploration Geophysicists, 2096–2100.

Xiong, W., X. Ji, Y. Ma, Y. Wang, N. M. AlBinHassan, M. N. Ali, and Y. Luo, 2018, Seismic fault detection with convolutional neural network: Geophysics, 83, O97–O103.

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

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

Yilmaz, O., 1987, Seismic data processing: Investigations in geophysics.

Zeng, Y., K. Jiang, and J. Chen, 2019, Automatic seismic salt interpretation with deep convolutional neural networks: Proceedings of the 2019 3rd International Conference on Information System and Data Mining, 16–20.

Zhang, H., P. Zhu, Y. Gu, and X. Li, 2019, Automatic velocity picking based on deep learning, in SEG Technical Program Expanded Abstracts 2019: Society of Exploration Geophysicists, 2604–2608.

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

Zhu, Q.-Y., A. K. Qin, P. N. Suganthan, and G.-B. Huang, 2005, Evolutionary extreme learning machine, Pattern recognition, 38, 1759–1763.