International Journal of Advances in Scientific Research and Engineering (ijasre)

E-ISSN : 2454-8006

DOI: 10.31695/IJASRE.2022.8.7.2

Volume 8, Issue 7
July - 2022

Study of Small Dataset of Images for Seismic Fault Imaging with a Supervised Convolutional Neural Network

Munezero Ntibahanana[1,2,3], Tondozi Keto[1,3], Moise Luemba[2,3], Raïs Seki Lenzo[1,3], Yannick Mananga Thamba[1,3], Kevin Lumpungu Lutumba[1], Yang'tshi Ndong Olola[3], Emmanuel Lokilo Lofiko[4] , Alidor Kazadi Mutambayi[1]

[1]Centre for Research in Geophysics (C.R.G.), Kinshasa, RD Congo

[2]School of Geosciences, China University of Petroleum (East China), Qingdao, China

[3]Faculty of Oil, Gas and Renewable Energies, University of Kinshasa, DR Congo

[4]Faculty of Sciences, University of Gbadolite, Nord-Ubangi, DR Congo

ABSTRACT

Recognizing faults in seismic images is crucial for structural modeling, prospect delineation, reservoir characterization, and well placement. Basically, faults have the appearance of lateral reflection discontinuities in seismic images and are interpreted using seismic attributes that measure those discontinuities such as coherence, and curvature. However, methods based on seismic attributes are often more challenging, time-consuming, and may suffer from noises and sensitivity of stratigraphic features, which also tie in reflection discontinuities. Therefore, we propose a solution for delineating faults from 3D seismic images using a supervised fully convolutional neural network (CNN). This approach uses a pixel-by-pixel prediction in 3D seismic images to classify whether a given pixel is a fault or a non-fault. The trained model learned to bank on rich and proper features that are important for the recognition of faults and achieved 97% of accuracy. To test the effectiveness of our model, we used new 3D seismic images, and the results displayed clean and accurate recognition of faults within only milliseconds, saving time and optimizing the accuracy. In this paper, we showed that by using only a few 3D seismic images from a given seismic volume to train the model, not only do we handle one of the difficulties encountered by researchers to obtain a sufficient amount of data needed to train common CNN models but also, interpreters can successfully predict faults in any other seismic image from the same volume.

Key Words: Binary Image Segmentation, Convolutional Neural Networks, Deep Learning, Seismic Fault Interpretation.

1. INTRODUCTION

It is known that subsurface geology modelling is very challenging and involves many tasks, such as faults interpretation, which is often based on seismic images. However, faults recognition in seismic images is challenging and requires considerable manual labour and time. Since faults are essential for the localisation of oil regions, prospect delineation, reservoir characterization, as well as well placement, researchers in exploration, development and production petroleum industry are continuously developing tools to enhance their recognition from seismic data. Basically, faults show high lateral discontinuities or low lateral continuities of reflections in seismic images [1]. Then, traditional interpretation method requires tracing and picking them manually but, this is still a time-consuming and a manual work intensive process and often results in very low interpretation efficiency. As for example, an interpretation expert may take from weeks to months to label faults within a typically sized seismic volume [2]. Taking the benefit of the way faults are viewed in a seismic image, geophysicists have made efforts to develop a suite of computer-aided tools, namely seismic attributes such as coherence, curvature and flexure attribute. The coherence attribute has been used [3] to estimate the lateral similarity of seismic reflections while the curvature attribute [4, 5] and the flexure attribute [6] has been used for the purpose of outlining small-scale structures such as subtle faults and fractures. Moreover, the coherence attribute has been proved effective in depicting faults and stratigraphic features and various cases of study has demonstrated the efficiency of curvature and flexure attributes to identify planar seismic features like fractures. Other seismic attributes such as the

semblance [7], the variance [8, 9] and the gradient magnitude attributes [10] has been invented to measure the seismic reflection continuities and discontinuities. Although these tools have brought relevant contributions in seismic interpretation to the extent that they are still widely used, it is worth mentioning that seismic attributes are limited and insufficient in fault recognition because faults are sensitive to noise and stratigraphic features, which also may correspond to reflection discontinuities in seismic images [11].

Nevertheless, semi-automatic fault extraction tools have been popular in the past years with numbers of algorithms presented in this field, including ant tracking [12], Hough transform [13], Eigenvector analysis [14], dynamic time wrapping [11], motion vector [15] and more, but also, it is observed that applying such algorithms on a seismic volume is a time-consuming process, particularly for large datasets since the classification have to be repeated at every sample in the volume [16]. However, with the recent success of machine learning in computer vision domain, many novel techniques are being introduced into the field of seismic interpretation in order to achieve better results on fault detection [17, 18, 19], salt-body delineation [20], facies analysis [21] and even in the interpretation of direct hydrocarbon indicator (DHI) characteristics [22]. Thus, the convolutional neural networks (CNN), which constitute a subset of machine learning techniques that belongs to the deep learning type of algorithms [23] have proven to be the most powerful method in solving problems of object detection, image classification and segmentation nowadays [24]. In this work, we implement a CNN model and investigate its effectiveness to recognize faults in a 3D seismic image. As it is customary into the training of common deep learning models, even if there is a need of huge amount of data to train a good CNN model but, we think that the interpreter may need only to pick and label a relatively small 3D seismic image dataset of a given seismic volume to train the network then, the trained model can accurately recognize and predict faults in any other image of the same seismic volume.

2. MATERIAL AND METHODS

2.1 Framework and Libraries

For the implementation of our CNN fault recognition model, we have used the following basic python frameworks and libraries: Tensorflow which is one of the most popular machine learning framework developed by the Google Brain team; Keras, an open-source library for deep learning that can be used on Tensorflow [25]; Numpy, penCv2 and Matplotlib. Aside from the above packages in the list, most of what we need comes preinstalled with Anaconda browser which is a data science platform having one of the best integrated development environments (IDEs) for python data science [26] like Jupyter Notebook that we have used to write our codes. But, usually, a predictive machine learning model need labels to be trained. To create faults labels from our training 3D seismic images, we have used GIMP software, which is free accessible online software.

2.2 CNN and Seismic Fault Recognition

2.2.1 CNN Model Theory

In computer vision, CNN models are now state-of-the-art among the most popular techniques used for image segmentation. In CNNs, fault recognition may be taken as a binary image segmentation problem where we can classify all pixels of a seismic image into two classes, namely: fault class and non-fault class. This classification task is achieved by labelling all pixels belonging to fault class with ones and all pixels belonging to non-fault class with zeros in a 3D seismic image. To illustrate the convolution layers, let us take a three-layer CNN as an example (Figure1). In this example, consider that each layer has one channel where w^l and b^l denotes the convolution filter and the bias in the kth layer. The ˆ and ¯ differentiate the convolution operations applied on different locations of the image. $\hat{w}^l = \overline{w}^l$; $\hat{b}^l = \overline{b}^l$ are referred to as weight sharing of the CNN so; the convolution layer can be expressed (Equation 1) as follows:

$$y^l = f(w^l \, X \, y^{l-1} + b^l) \qquad (1)$$

Where X is the convolution operation, f the nonlinear activation function; w^l and b^l are respectively convolution kernel and bias for layer l. Often, a convolution layer could have many kernels $(w^{l,k}, k = 1, \dots, K)$ and will result in output feature of K channels. When w^{l-1} has C channels, the value in the kth channel and position (i, j) of the output the (Equation 2) in layer l is given by:

$$y^l_{k,i,j} = f\left(\sum_{c=1}^{C} \sum_{m=1}^{M} \sum_{n=1}^{N} w^{l,k}_{c,m,n} \, y^{l-1}_{c,i-\left[\frac{M}{2}\right]+m,j-\left[\frac{N}{2}\right]+n} + b^{l,k}\right) \qquad (2)$$

In this equation, $w^{l,k}$ is the designed kth kernel of size [C, M, N] indicating the channel, the height and the width of the kernel. The kernel should have the same channel as w^{l-1}. To sum up, the main prediction CNN model may consist of several convolution blocks [27], where each block consists of:

DOI: 10.31695/IJASRE.2022.8.7.2

- Convolution layer (Conv),
- Dropout layer placed immediately after the Convolution layer,
- Rectified linear unit (ReLU) layer as nonlinear activation function,
- Max pooling layer (Pool) and,
- Fully-connected layer.

In a binary segmentation task, the fully connected layer has a sigmoid activation function, which allows the output of the CNN model to be treated as a probability distribution across two classes (Figure 2), where 0 denotes the non-fault class and 1 the fault class.

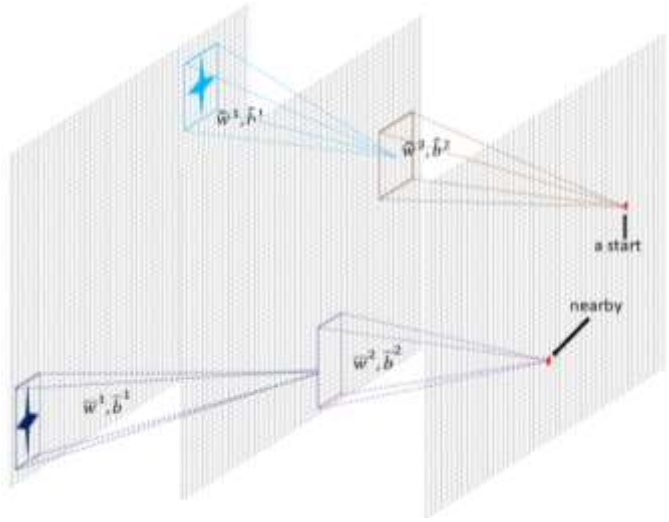

Figure 1.An illustration of convolution layers. The kernel and bias in each layer are weight sharing.

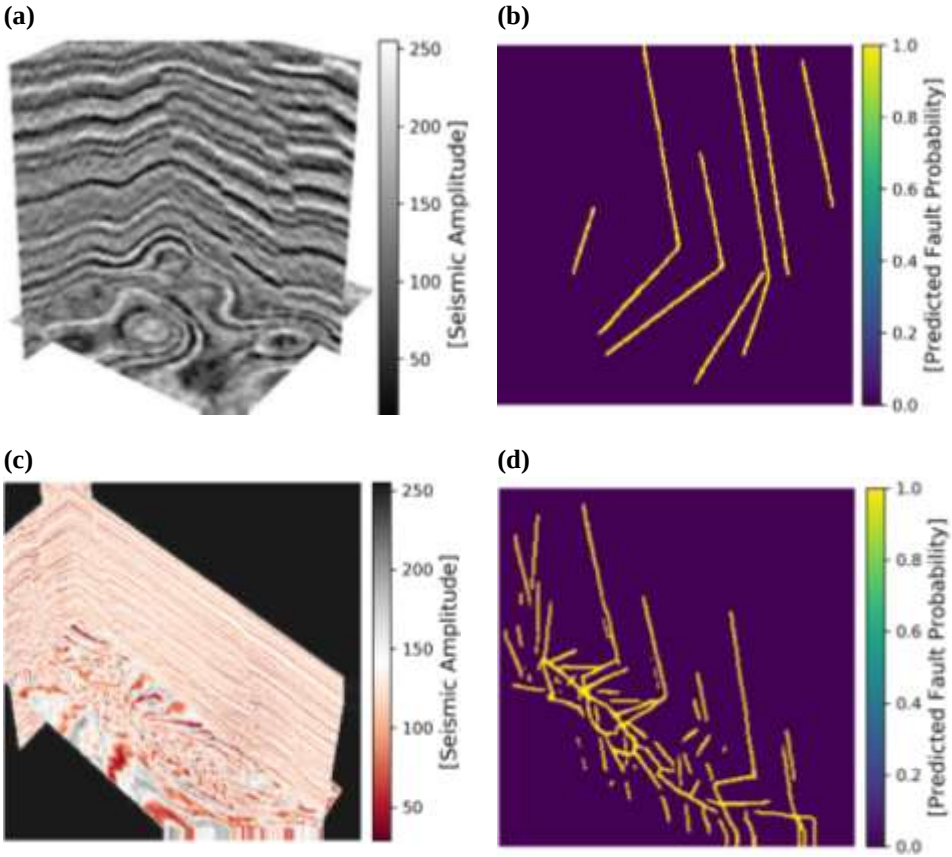

Figure 2. Examples of predicting3D seismic images (a) and (c) and corresponding predictions of fault probability maps (b) and (d). We can see clearly that all probabilities range from o to 1 and all zeros represent the non-fault class while all ones stand for fault features.

DOI: 10.31695/IJASRE.2022.8.7.2

More clearly, if we assume that $I_i \in R^{m_i \times n_i \times c_i}$ is the output of layer i of the CNN with dimensions of $m_i \times n_i \times c_i$ channels, the relationship between the output I_i^j of a single channel j of layer i (called a feature map) and the output of the previous layer can be given by the relation below (Equation 3) [28] where $h \times I$ is a 2D convolution operation of an image I with a filter h, b is a bias and σ is a ReLU function.

$$\mathrm{I}_i^j = \sigma\left(\sum_{k=1}^{c_{i-1}} h_{ijk} \times \mathrm{I}_{i-1}^k + b_i^j\right) \quad (3)$$

In learning operations, CNN models expect the spatial relationship between pixels by learning internal feature representations using small squares of input images and preserve those relationships. For a given image, features are learned and used across the whole image, allowing objects in the image to be shifted or translated in the scene while still detectable by the model. CNN models use fewer weights to learn and remain unchanged to object position and distortion in the scene. CNN models automatically learn and generalize features from the input domain.

2.2.2 CNN Architecture

Based on the type of task to be accomplished by the predictive CNN model, there are various ways of designing the architecture of model. Here, we have chosen to use a modified version of the Unet architecture (Figure 3) which is a supervised end-to-end fully CNN model. The original Unet [29] was proposed for biomedical image segmentation problems. Nowadays, Unet is widely used for many other image segmentation problems.

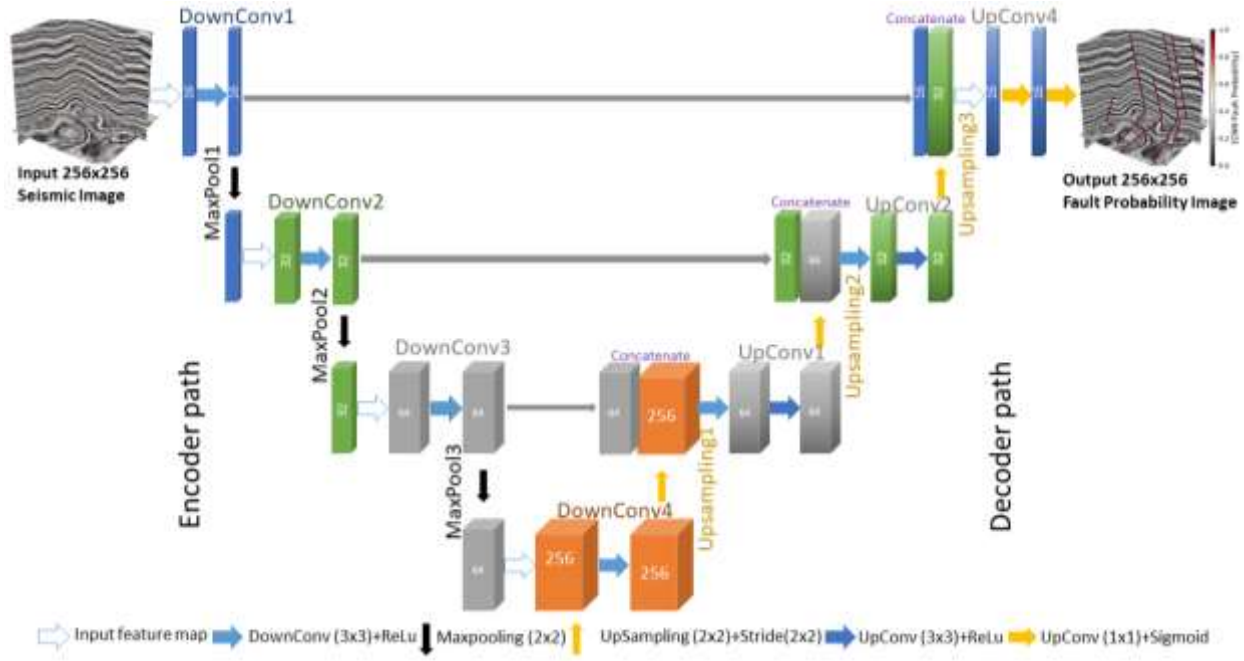

Figure 3. A modified Unet architecture constituting our model that we used for seismic fault recognition: an input 256 x 256 x 3 seismic image is given to encoder path of the model and goes through convolution layers to be down-sampled then, passes to the decoder path for the up-sampling operation. We apply a sigmoid function to the final feature map in order to get the fault probability map of the same size as the input seismic volume.

Like in the original Unet, our modified architecture consists of both a contracting path for the down sampling of the image (encoder path) and a symmetrical expansive path for the up sampling of the image (decoder path). However, we have reduced the number of convolution layers from twenty-three in the original Unet to fifteen in order to save the computation cost while preserving good performance in fault segmentation at the same time. In addition, the number of features is consequently reduced proportionally to the number of convolutional layers.

2.2.2.1 Operations in Encoder Path

In the encoder path, each step has two 3×3 convolutional layers respectively followed by a ReLU and a 2×2 max pooling operation with stride 2 for down sampling. To prevent the overfitting or the under fitting during the training, we defined a dropout layer right after each convolution and before the pooling operation and we doubled the number of features after each step.

DOI: 10.31695/IJASRE.2022.8.7.2

2.2.2.2 Operations in Decoder Path

In the decoder path, every step is made by a 2x2 up sampling operation, a concatenation operation with features from the encoder path, and two 3x3 convolutional layers followed by a ReLU. The up-sampling operation is implemented using the function Conv2DTranspose defined in keras [30] and the final output layer is a 1x1 convolutional layer with a sigmoid activation such as logistic function (Equation 4) to map each final feature vector into a probability value for the output fault probability map. Herein, x_0 is the x value of the midpoint of sigmoid, L the curve's maximum value, k the logistic growth rate or steepness of the curve. For values of x in the domain of real numbers from $-\infty$ to $+\infty$, the S-curve shown on [31] is obtained, with the graph of f approaching L as x approaches $+\infty$ and approaching zero as x approaches $-\infty$.

$$Sigmoid(x) = \frac{L}{1+e^{-k(x-x_0)}} \qquad (4)$$

2.2.3 Train and Validate the Model

2.2.3.1 Data Pre-Processing

The pre-processing of the training data is crucial to train a good CNN model. In this stage, we increased the diversity of the training dataset across the sampled seismic volumes. This diversity increasing serves to prevent the model from learning irrelevant features. Herein, we have performed a data augmentation processing using the modified Keras Image Data Generator API [32] where each of the 6 well collected 3D seismic images as well as their corresponding fault labelled images was set to be converted to grayscale, resized to a resolution of 256X256X1 (for fault label images) and 256X256X3 (for seismic images), then we rotated, zoomed in and flipped all the images so that the final results allowed us to generate the 672 pairs of seismic and binary fault labelled images that we used to train and validate our CNN model.

2.2.3.2 Training and Validation

To train and validate the CNN model, we have split all the prepared 672 pairs of 3D seismic and binary fault label images into training and validation sets. Herein, we took the 86% of the dataset for training purpose while the remaining 14% of the same dataset is set aside to be used for the validation purpose. All images have been fed into the model in batches of 16 images. We also applied the Adam function to optimize the model parameters and we defined the number of epochs to be 150. At every epoch, all the training images are processed completely.

2.2.3.3 Training and Validation Losses

In order to measure the error of fault recognition process, we have used the cross-entropy loss function (Equation 5). This loss function is widely used for binary segmentation of common images. As the labels are all binary values (ones or zeros), the first term measures the prediction error at the image pixels labelled by ones while the second term measures prediction error at the pixels labelled by zeros.

$$L = -\sum_{i=0}^{i=N} y_i \, log(p_i) - \sum_{i=0}^{i=N} (1 - y_i) log(1 - p_i) \quad (5)$$

In this equation N is the number of pixels of the input image; y_i is true binary labels and p_i is the prediction probabilities ($0<p_i<1$) computed from the sigmoid activation in last convolutional layer. As shown below (Figures 4 and 5), after 150 epochs, the trained CNN model qualitatively achieved 97% of accuracy and 6, 5% of loss in training while the validation accuracy and loss respectively displayed, 96, 8% and 7%.

DOI: 10.31695/IJASRE.2022.8.7.2

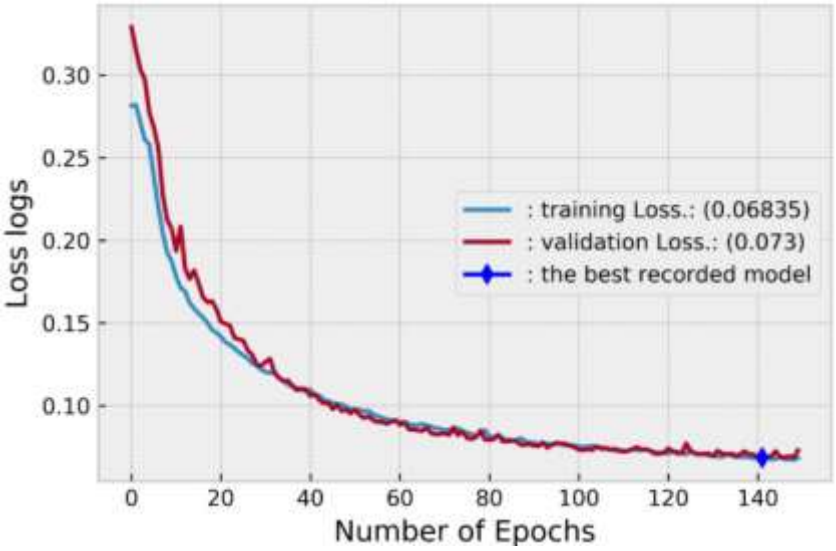

Figure 4. Training and the validation losses decrease along with the number of epochs. The best recorded model was observed after 140th epoch on the validation curve, converging to 6,9 % of loss

Accuracy metric computes how the predictions equal training labels. It creates two local variables which are the total and the count used to calculate the frequency with which predicted labels match the training ones. This frequency is ultimately returned as binary accuracy: an idempotent operation that simply divides total by count [33]. In short, the accuracy metric tells us what % of predictions was correct. In summary, the training and validation process lasted nearly 6 hours but could be much lower by using a high-performance computer.

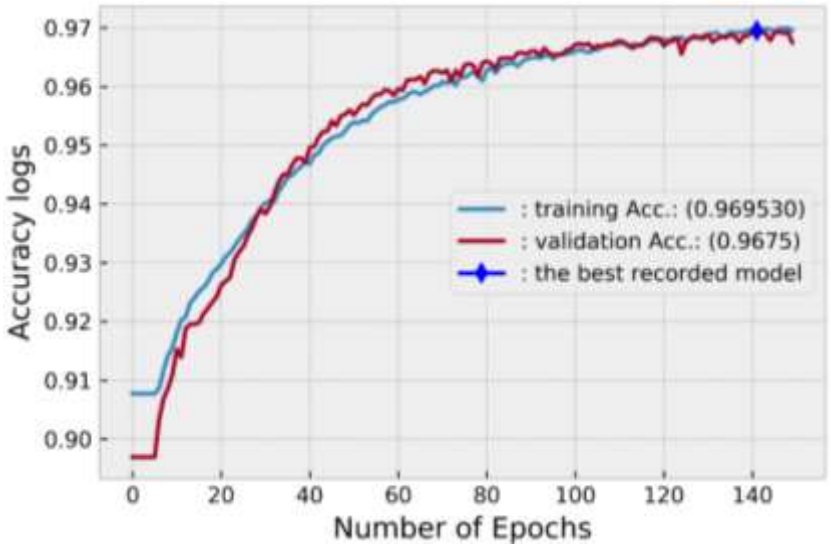

Figure 5.Training and validation accuracies increase along with the number of epochs. The best recorded model was observed after 140th epoch on the validation curve, converging to 96.99 % of accuracy

2.2.3.4 Model Performance Evaluation

Apart from the above built-in accuracy metric function; to evaluate the performance of our trained model, we defined two types of metric function for evaluation: the classification and segmentation metrics. For segmentation evaluation, the mean Intersection-Over-Union (mean IoU) metric function has been calculated and the precision-recall, the true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN) numbers intervened in order to quantitatively estimate fault and non-fault classification performance of our trained model.

- *Mean Intersection-Over-Union:*

The mean Intersection-Over-Union (mean IoU) is a common semantic image segmentation metric often used for the evaluation purpose. It first calculates the IoU (Equation 6) for each existing semantic class and then gives the average over those classes [34]. This metric is computed based on TP, FP and FN numbers defined in the paragraph above.

DOI: 10.31695/IJASRE.2022.8.7.2

$$IoU = \frac{TP}{TP+FP+FN} \quad (6)$$

- *Precision-recall metric:*

Precision and recall metrics are calculated based on the TP and FP variables (Equations 7 and 8). These metrics respectively give the precision and the recall of the predictions with respect to the labels and are defined as follows:

$$precision = \frac{TP}{TP+FP} \ (7); recall = \frac{TP}{TP+FN} \quad (8)$$

The precision metric gives idea of what % of faults are correctly predicted by the model while the recall indicates what % of true faults the model detected.

- *TP, TN, FP and FN rates:*

True positives (TP), false positives (FT); true negatives (TN) and false negatives (FN) rates [35] are metrics used to calculate four variables that keep tracking the accumulated number of the true faults predicted as faults, the false faults predicted as faults, the false faults predicted as non-faults and the true faults predicted as non-faults [36] respectively. These rates constitute a classification performance metric that is especially useful for imbalanced class problems [37] as it is the case in our fault segmentation task where we may have more than 90 % as non-fault features in seismic images.

3. RESULTS AND DISCUSSIONS

3.1 Results

Recall that we have been implanting a CNN model in order to predict whether pixels belong to fault class or to non-fault class within a given 3D seismic image. Therefore, for each pixel we got a value between 0 and 1 after the training step where 0 represents the non-fault class and 1 represents the fault class. As we got probability values from 0 to 1, therefore to display the prediction results, we set 0.5 as the threshold above or under which to decide whether to classify a pixel as 0 or 1. Below we present some samples of the prediction results and we compare them to the pre-computed classification and segmentation metrics. For the classification metrics, the fault classification results are shown (Figure 6) and the effective fault recognition on two different seismic volumes (Figure 7). Ideally, a good model should have high TN and TP than the FP and FN [36]. When comparing these parameters from the charts in figure 6 and 8, we can observe that our CNN model achieved high TN and TP, meaning that in its prediction task, a very good proportion of false faults have been detected as non-faults and a lot of true faults has been recognized as true faults in both training and validation data sets.

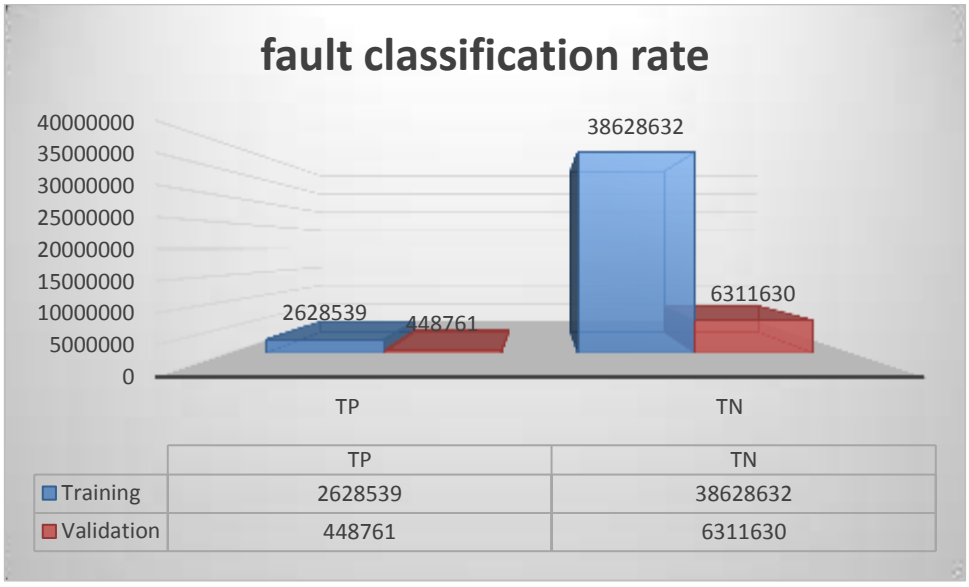

Figure 6. From this chart we have observed that many false fault features have been detected as false allowing the model to distinguish them from those of the true faults. Then, comparing the TN and TN we realise that our CNN performed good prediction of true faults during the training and validation steps.

DOI: 10.31695/IJASRE.2022.8.7.2

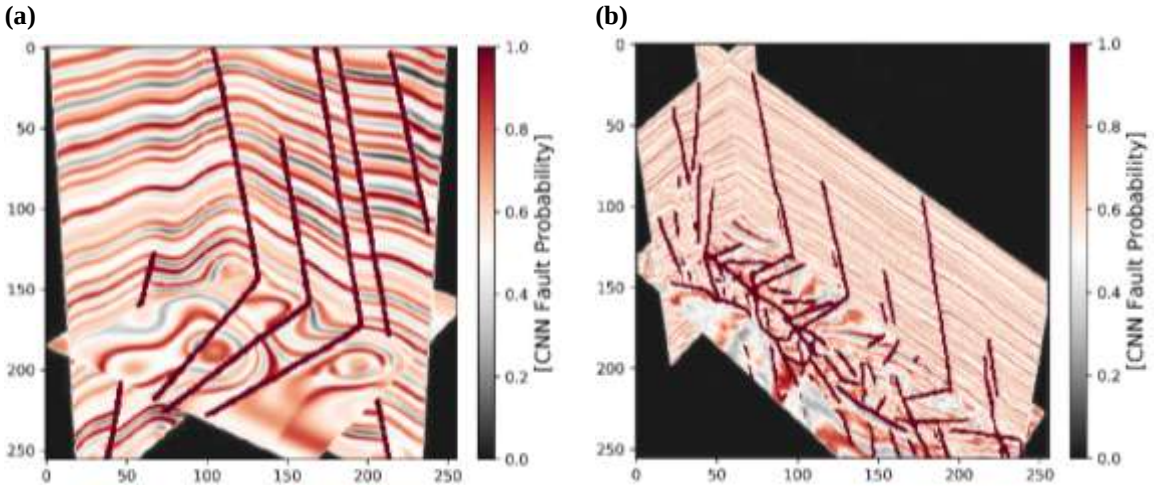

Figure 7. Illustration of fault segmentation performance on random samples: (a) is from the training dataset and (b) is from the validation dataset.

One may need to get more information about the non-fault or the false fault rates that the model has predicted. For that purpose, we used the precomputed FP and FN parameters as shown in the chart below (Figure 8).

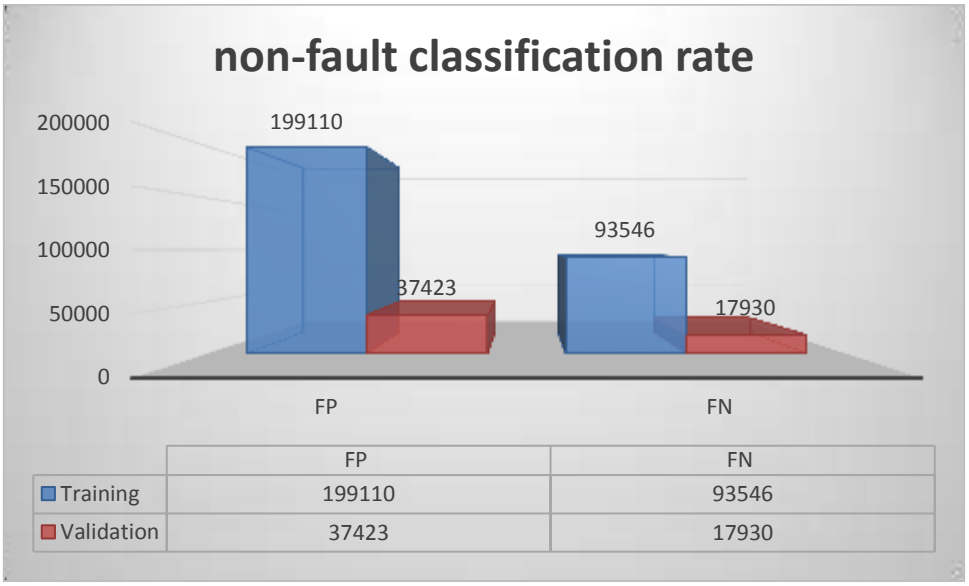

Figure 8. Non-fault classification metric of our trained model. When comparing this chart to the chart in figure 6, we can see that insubstantial non- fault features are detected as faults. It means that our model is recognizing faults very well in both training and validation data sets.

FP is computed to show what proportion of non-fault features that our CNN model predicted as fault features. As observed in this chart (Figure 7) and when comparing it to the chart in figure 6, for both training and validation data, the model has recognized an insubstantial number of non-fault features taken for faults features. By looking at some other results below (Figure 9), we can notice for a given and complete seismic volume, it may be more likely that there may be many faults that interpreters cannot see. In fact, based on some subjective reasons, during the manual labelling of faults to create the training fault labels, we may not label each of them; so, some relatively small and non-obvious faults may have been excluded. Consequently, the model may miss some faults on one hand and make mistakes in prediction when the prediction image is containing complicated fault features and prediction results may exclude the relatively small faults. Therefore, we have calculated the precision recall metric to appreciate how our model is performing in this case.

DOI: 10.31695/IJASRE.2022.8.7.2

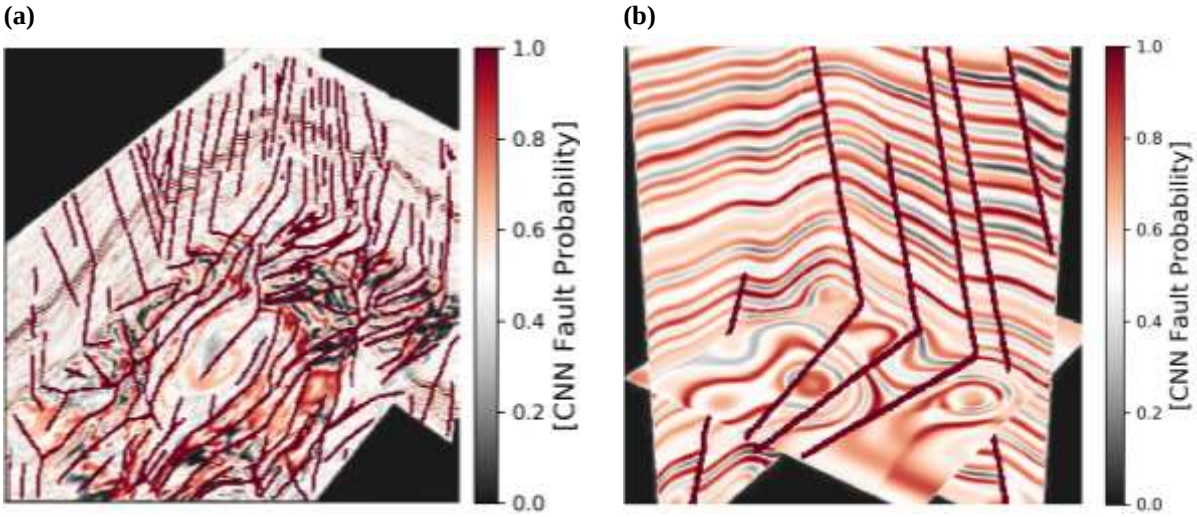

Figure 9. Faults are recognized in a complicated 3D seismic image (a) while the recognition is better, clean and accurate for not complicated 3D seismic image (b).

The precision metric gives the idea of what percentage of faults are correctly predicted by the model while the recall indicates what percentage of true faults the model detected. As shown in the above chart (Figure 10), our CNN model shows more or less the same values for both training and validation data sets meaning that all faults that the model predicted are all true faults.

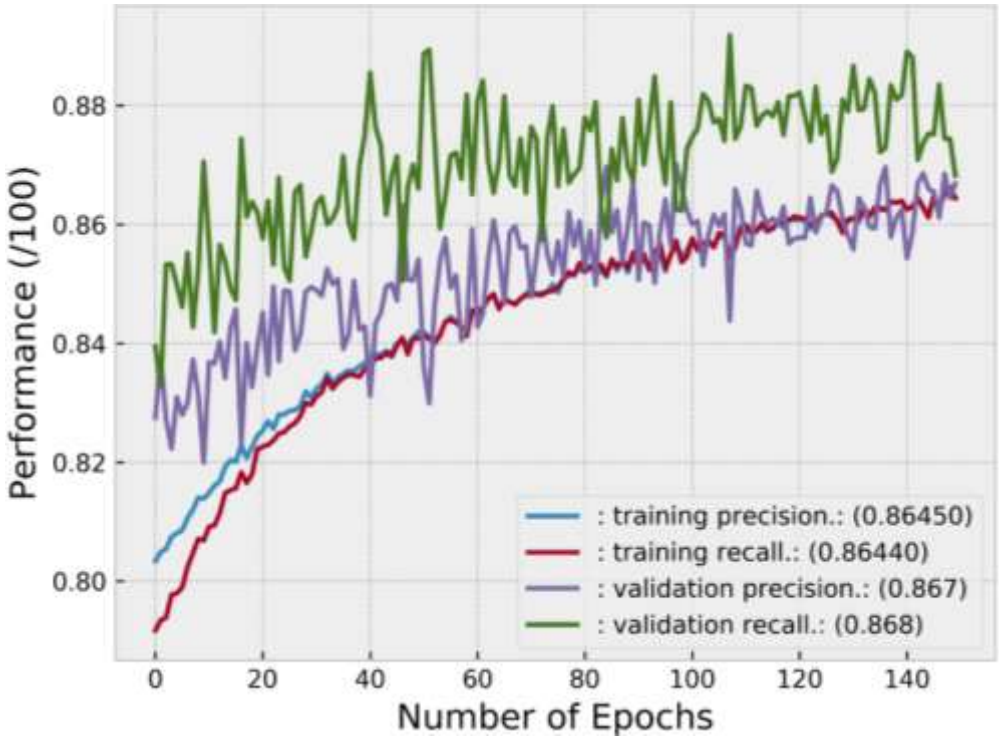

Figure 10. Precision-recall plot is used to evaluate the fault detection and prediction on both training and validation data sets. Results show that our CNN performed very well in fault recognition as.

In the chart above, we can see that 86,7% of the validation predictions are correctly classified as true faults from 86,8% of actual true fault features. However, we need to know how well our model can work on unseen data to recognize fault features. For this purpose, we have performed the model test. In this stage, the testing images are all 3D seismic volumes that have not been used, neither into training process, nor in validation one. They are considered as new data only used to test the relevance of our model. Results below (Figure 10) have confirmed that our model is performing well enough in the recognition of faults from other 3D seismic images than those used to train and validate the model.

DOI: 10.31695/IJASRE.2022.8.7.2

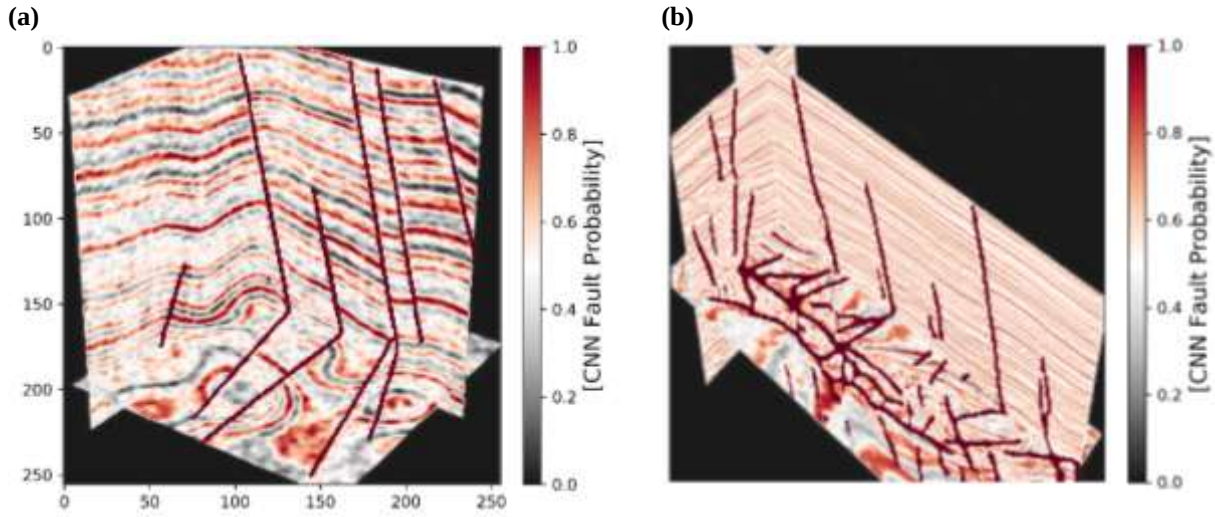

Figure 11. Faults recognition on the testing images; (a) shows accurate and clean recognized faults (b) also shows good results but we can still be viewing same ambiguities on the time slice probably due to its complicated nature in fault features displaying.

To give an idea of how qualitatively the model works in fault recognition on the testing and unseen data, we have reported (Figure 11) the mean values of the calculated segmentation metric for both training and validation data.

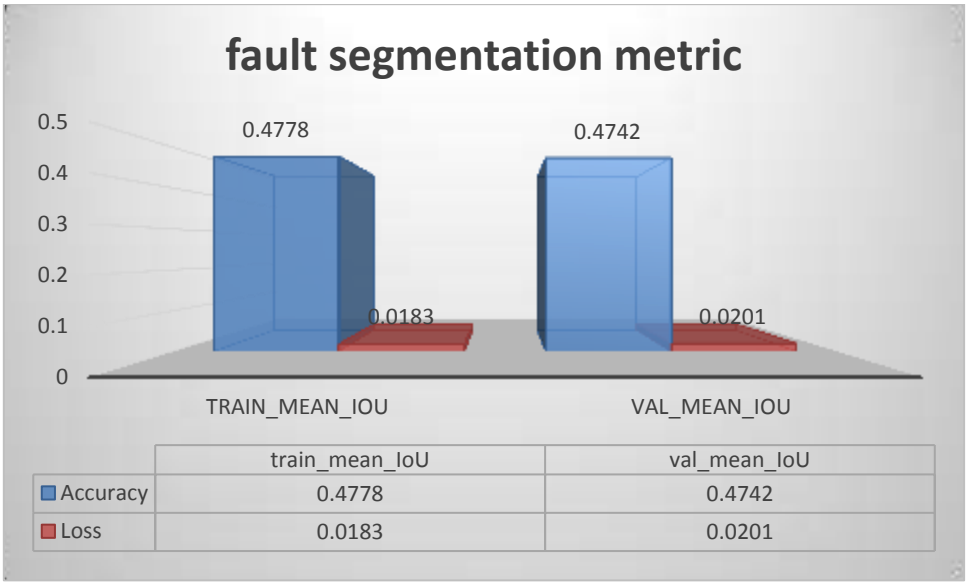

Figure 12. Our CNN model shows more or less the same values for both training and validation data sets. According to the observed mean IoU values our CNN model can realize good segmentation performance on both training and validation data sets.

The Intercept over Union (IoU) segmentation metric reflects the correlation between the ground truth and prediction. The higher is the IoU value, the higher is the correlation between the ground truth and prediction and vice versa.

3.2 Discussions

In this research, we are investigating the possibility of using a small training image set to train a CNN model for 3D seismic fault recognition. Since we have used a small amount of data as the training set (only 672 pair of images), the generalization performance of the model could be sacrificed. That means, our model may not make good predictions on other volumes, which are not acquired from the same area of the selected training samples. However, this does not affect the practical application of the proposed model. In fact, the purpose of our research was not to address the model generalization issues that we will consider in another research.

As the distribution of fault features depended on the geology under the study region, the proposed method will perform well when the testing or predicting data and training data come from the same seismic survey. That is why the performance cannot be guaranteed when they are from different volumes. Indeed, by simply labelling several seismic sections, from different surveys we can obtain more accurate predictions on most seismic images from different surveys. However, it's often difficult to obtain a large

DOI: 10.31695/IJASRE.2022.8.7.2

amount of labelled real data which are required to train a very good CNN model. Consequently, synthetic data are usually used as training sets in most cases but models trained by only synthetic data may results in a poor performance. For example, when we are predicting land data having much lower signal-to-noise ratios, whereas the use of real data to train networks should result in predictions that are more accurate. Thus, with our proposed method we can use real data as the training set without any concerns about insufficient data and achieve good results in fault recognition.

To train the model, interpreters need only to pick and label a few 3D seismic images, which is very easy to accomplish and with image data augmentation tips, they can generate enough data for training. Nevertheless, if we want to enhance the generalization performance of our trained model, we can use the strategy of transfer learning by labelling some other samples from the same or a new seismic volume to increase the diversity of the training dataset.

4. CONCLUSIONS

We have implemented a supervised fully CNN model to efficiently and accurately detect faults in 3D seismic images. This was possible even using a small training dataset. The fault recognition was performed as a binary segmentation problem. We used a modified Unet architecture to save the computational cost while keeping to preserve good performance and, the trained model achieved good enough performance in detection of faults from 3D seismic images when looking at both the prediction results and values of the evaluation metrics. Although the model was trained using few fields seismic image collected from different surveys, the testing results showed that fault recognition works very well for any 3D seismic image from the same surveys. After the model has been well trained, to predict faults in a 256x256x256 sized seismic image, our trained model takes only milliseconds and the displayed results after prediction are cleaner and more accurate than those of any other conventional fault detection method. At the same time, the application of this model is of interest with a double advantage: saving the operational time and maximizing the accuracy of faults interpretation in 3D seismic images.

ACKNOWLEDGEMENT

The authors thank the department of geophysics of China University of petroleum (East China) for the support in terms of facilities and all anonymous readers for helping to improve the manuscript.

REFERENCES

[1] W. Xinming, L. Liang, Y. Shi and S. Fomel, “FaultSeg3D: using synthetic datasets to train an end-to-end convolutional neural network for 3D seismic fault segmentation,” *Geophysics,* vol. 84, no. 3, pp. IM35-IM45, 2019.

[2] S. Li, Y. Changchun, S. Hui and Z. Hao, “Seismic fault detection using an encoder-decoder convolutional neural network with a small trainingset,” *Journal of Geophysics and Engineering,* pp. 1-15, 2019.

[3] Bahorich and Farmer, “3-D seismic discontinuity for faults and stratigraphic features: The coherence cube,” *The Leading Edge,* vol. 14, pp. 1053-1058, 1995.

[4] A. Roberts, “Curvature attributes and their application to 3D interprated horizons,” *First Break,* vol. 19, pp. 85-100, 2001.

[5] T. Boe and A. a. Daber, “Seismic features and the human eye: RGB blending of azimuthal curvatures for enhancement of fault and fracture interpration,” in *SEG Technical Program*, 2010.

[6] D. Gao, “Integrating 3D seismic curvature and curvature gradient attributes for fracture detection: Methodologies and Interpretational implications,” *Geophysics,* vol. 78, no. 2, pp. 021-038, 2013.

[7] K. J. Marfurt, K. R. L., F. S. L. and B. and M. S., “3-D seismic attributes using a semblance-based coherency algorithm,” *Geophysics,* vol. 63, pp. 1150-1165, 1998.

[8] V. P. Bemmel and R. Pepper, “Seismic signal processing method and apparatus for generating a cube of variance values”. Washington DC. Patent 6,151,555, 2000.

[9] T. Randen, S. Pedersen and L. Sonneland, “Automatic extraction of fault surfaces from three-dimensional seismic data,” in *71st Annual International Meeting, Expended Abstracts*, 2001.

DOI: 10.31695/IJASRE.2022.8.7.2

[10] A. Aqrawi and T. Boe, "Improved fault segmentation using a dip guided and modified 3D Sobel filter," in *81st Annual International Meeting*, 2011.

[11] D. Hale, "Methods to compute fault images, extract fault surfaces, and estimate fault throws from 3D seismic images," *Geophysics*, vol. 78, pp. 33-43, 2013.

[12] S. I. Pedersen, T. Randen, L. Sonneland and Steen, "Automatic fault extraction using artificial ants," in *72nd Annual International Meeting*, 2002.

[13] N. Al Bin Hassan and K. Marfurt, "Fault detection using Hough transforms," in *73rd Annual International Meeting*, 2003.

[14] A. Barnes, "A filter to improve seismic discontinuity data for fault interpretation," *Geophysics*, vol. 71, no. 3, pp. 1-4, 2006.

[15] Z. Wang, Z. Long, G. AlRegib, A. Asjad and M. Deriche, "Automatic fault tracking across seismic volumes via tracking vectors," in *IEEE International Conference on Image Processing*, 2014.

[16] D. Haibin, W. Zhen and A. Ghassan, "Real-time seismic image interpretation via deconvolutional neural network," in *SEG International Exposition and 88th Annual Meeting.*, 2018.

[17] Z. Zheng, P. Kavousi and H. Di, "Multi-attributes and neural network-based fault detection in 3D seismic interpretation," *Advanced Materials Research*, pp. 1497-1502, 2014.

[18] G. Huang, Z. Liu, L. VanDerMaaten and K. Weinberger, "Densely connected convolutional networks," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[19] H. Di, M. Shafiq and G. AlRegib, "Seismic fault detection based on multi-attribute support vector machine analysis," in *87th Annual International Meeting*, 2017.

[20] Y. Alaudah and G. AlRegib, "A weakly-supervised approach to seismic structure labeling," in *87th Annual International Meeting*, 2017.

[21] Z. Tao, Q. Jie, L. Tengfei, L. Fangyu and M. Kurt, "Semisupervised multiattribute seismic facies analysis," *Interpretation*, vol. 4, no. 1, pp. SB91-SB106, 2016.

[22] R. Rocky and C. ChingWen, "Interpretation of DHI characteristics with machine learning," *Geophysical Insights*, vol. 35, no. 5, pp. 55-63, 2017.

[23] B. Jason, Deep Learning with Python, Develop Deep Learning Models on Theano and TensorFlow using Keras, v1.18 ed., Melbourne: Machine Learning Mastery, 2019.

[24] W. Xinming, Y. Shi, F. Sergey, L. Luming and Uber, "Convolutional neural networks for fault interpretation in seismic images," in *SEG International Exposition and 88th Annual Meeting*, 2018.

[25] S. Hyunseok, K. V. V. Masoud Badiei, H. Charles, R. Honyi, X. Ruoxiu, J. Xiao and X. Lei, "Machine Learning Techniques for Biomedical Image Segmentation: An Overview of Technical Aspects and Introduction to State-of-Art Applications," *Medical Physics*, 2020.

[26] S. Himanshu, Practical Machine Learning and Image Processing for Facial recognition, Object Detection and Pattern Recognition Using Python, Allahabad, Uttar Pradesh, India: Apress, 2019.

[27] A. Antoly, D. Andrew and F. Di, "Application of deep learning for subsurface faults detction with seismic data," *CS230, Stansford University, CA*, 2019.

[28] V. Nair and G. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, Haifa, Israel, 2010.

DOI: 10.31695/IJASRE.2022.8.7.2

[29] O. Ronneberger, P. Fischer and T. Brox, "U-Net: convolutional networks for biomedical image segmentation," *Medical Image Computing and Computer-Assisted Intervention-MICCAI*, pp. 234-241, 2015.

[30] D. Vincent and V. Francesco, "A guide to convolution arithmetic for deep learning," 12 January 2018. [Online]. Available: https://arxiv.org/abs/1603.07285. [Accessed 19 September 2020].

[31] Wikipedia, "Logistic function," 2020. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Logistic_function&oldid=977583860. [Accessed 9 September 2020].

[32] R. Adrian, Keras ImageDataGenerator and Data Augmentation, pyimagesearch, 2019.

[33] Keras.a, "Accuracy metrics," 2020. [Online]. Available: https://keras.io/api/metrics/accuracy_metrics/#accuracy-class. [Accessed 13 October 2020].

[34] Keras.b, "Image segmentation metrics," 2020. [Online]. Available: https://keras.io/api/metrics/. [Accessed 16 September 2020].

[35] Keras.c, "Classification metrics," 2020. [Online]. Available: https://keras.io/api/metrics/classification_metrics/classification-metrics-based-on-truefalse-positives-amp-negatives. [Accessed 16 September 2020].

[36] S. Manomar, Mastering Machine Learning with Python in Six Steps: a practical implementation guide to predictive data analytics using python, Bangalore, Karnataka, India: Apress, 2017.

[37] R. Sebastian and M. Vahid, Python Machine Learning, Second Edition ed., MUMBAI, BIRMINGHAM : Packt Publishing, 2017.

DOI: 10.31695/IJASRE.2022.8.7.2