Seismic Fault Detection

Aditya Shah, Ashwini Akula, Avisha Singh, Jake Joiner, Nivedita Shanbag, Vandan Pandya

Abstract

Traditionally, fault images are calculated from seismic data, and fault detection is performed using a fault enhancement algorithm. There is a lot of dependence on parameters in traditional methods, which means that the parameters have to be adjusted several times to achieve optimum detection. To avoid the strainful process, many deep learning techniques are applied for the detection of faults in the petroleum industry. Through this paper, we are attempting to do a comparative analysis of three such deep learning techniques: FPN, Unet, and Linknet along with three feature extraction techniques Resnet, Inceptionnet, and VGG16. The analysis is done in two steps: training and testing. In the training part, we train our model using the three image segmentation techniques along with each of the three feature extraction techniques. In the testing part, we will apply the trained model on a test image. At the end, we concluded with best segmentation technique and feature extractor on our dataset.

1. Introduction

Seismic fault detection means to detect a discontinuity or abruption in reflection in a seismic image. The interpretation of conventional faults is based on identifying discontinuities or abruptions in reflection images and tracking these faults on 2D or 3D seismic images, which is very operator-intensive and has a strong reliance on interpreters' experience. Eigen structure coherence, gradient structure tensor has noise, which is sometimes difficult to differentiate. Overall, traditional approaches used to detect faults were by looking at faults, use image processing, and using domain knowledge to interpret it. In order to solve the problem, we have utilized computer vision techniques to automate the process.

1.1. Literature Review

In [1], the authors developed a novel classification technique by which seismic faults can be detected more easily. The technique was the integration of computationally efficient Support Vector Machine (SVM)/ Multilayer Perceptron (MLP) algorithms and local seismic attribute patterns which is termed super-attribute-based classification. The technique is a five-step process involving steps of attribute selection, labeling, retrieving super attributes, training the SVM and MLP classifiers, and volumetric processing. MLP fault volume was used for showing how the technique improves seismic fault interpretation in Volumetric Fault Imaging, Automatic Fault Extraction, and Seeded Fault Picking. Convolutional Neural Network was also tested for automatic attribute extraction by the authors by using the convolutional layers as an adaptive attribute generator. The authors achieved an accuracy of 83% by applying the super-attribute-based MLP and an accuracy of 88% by applying the trained MLP model from automatically extracted attributes by CNN.

The authors in [2] aim to skip the demanding and expensive steps of adjoint modeling and subsequent interpretation. This is performed using a deep neural network model which learns the relation between the data space and the final output. The output is handled by the Wasserstein loss function. The authors generate wavefields and record them as time-series signals based on an acoustic approximation to the wave equation. Many samples of the fault locations are obtained using this method and the corresponding synthetic data is generated. The synthetic data is then trained using DNN. The output of the DNN is a 3D voxel grid, the value of which depicts the likelihood of a fault is present in the specific voxel. Intersection over Union and Area Under the ROC metrics were used to measure performance. However, the scalability of this approach is yet to be analyzed.

A technique that makes use of the Convolutional Neural Networks (CNN) is developed in [3] for the automated detection and mapping of fault zones by making use of 3D seismic pictures. The technique comprises of two steps namely training and prediction. In the first step, the CNN model is trained with seismic picture cubes of field data with each one of them labeled as fault or non-fault. In the second step, the model is applied to predict fault likelihoods at the other locations in other seismic picture cubes. Classification accuracy of 73% was obtained on the training and validation datasets. For a real data cube of dimensions 1000X655X1083, a classification accuracy of 74% was achieved using the CNN model.

The author of [4] looks into the use of convolutional neural networks (CNNs) for seismic fault detection. They created a fictitious data set using basic fault geometry. The seismic amplitude is the only input to our network; the approach does not need to compute

any seismic attributes. They use a patch classification technique to classify the photos, which requires only a simple post-process to retrieve the precise defect site. On synthetic data, their network performs well, and when tested on portions of a genuine piece of the Netherlands offshore F3 block in the North Sea.

The authors of this paper [5] offer a method for detecting seismic faults using a CNN that only requires a minimal training set. They consider fault identification as a semantic segmentation problem and train a U-Net, an encoder-decoder CNN, to conduct pixel-by-pixel prediction on the seismic section to decide whether each pixel is a fault or not. They get decent prediction results on actual data using this form of CNN in the trials. When using the suggested technique to interpret a new seismic volume, interpreters just need to choose and identify certain 2D portions; the model can then anticipate faults in any other part of the same volume, considerably enhancing interpretation efficiency. They provide a defect detection accuracy index that describes the accuracy of the prediction findings to evaluate the suggested method's performance. They show in this study that by training a CNN with only seven seismic sections from a seismic volume, we can accurately forecast faults in any other section of the same volume.

By merging many features into a single object-sensitive property, artificial neural networks have been effectively used in seismic object recognition. This paper [6] explored multiple techniques. A case study from Stratton Field in south Texas, United States, demonstrated an efficient neural network defect detection technique. The new "fault probability" characteristic appears to be able to attenuate background noise while highlighting problems. The fault cube produces more convincing findings than individual qualities when ant-tracking is used. The structural complexity is also increased by multiple possible cross-strike fractures. The use of neural networks to identify faults aids structural interpretation and is important in hydrocarbon exploration, particularly in areas with complicated fault systems.

1.2. Business/ Analytics Problem and Question Framing

From the perspective of business value, our goal is to make fault detection more time-efficient and cost-effective. In turn, which will help in reducing the drilling time along with increasing safety. Leading to improved shipping and transportation.

1.3. Objective

The objective is to detect faults in seismic images using deep learning approach of image segmentation. Through this project, we are attempting to do a comparative analysis of three such image segmentation techniques: FPN, Unet, and Linknet along with three feature extractors Resnet, Inceptionnet, and VGG16. Leveraging best performing model, we are detecting faults in form of masked images.

1.4. Impact and Value

The image segmentation of the seismic data will help in pointing out the faults in the data. Moreover, it will also expedite the process of finding oil, confirming reserves of oil for loan acquisition, and drilling planning through reservoir modeling. Hence, from a business value point of view, the detection of faults will also reduce the time spent on all of the above activities thereby, making it time-efficient and cost-effective.

2. Data

2.1. Dataset background and quality

The dataset used in our project was the “Netherlands f3 block data”. It contains different images of synthetic data and segmented data. The dataset is a labeled one with a total of 400 images. Out of the 400 images, 200 are of synthetic seismic images and the rest 200 are the labeled images that clearly depict the faulty seismic data.

2.2. Data Augmentation

The reason for performing data augmentation on the dataset was that since the dataset consisted of only 200 synthetic seismic images, it would result in high loss and low validation mean IOU. Data augmentation acts as a regularizer to avoid overfitting. For our data set, we performed augmentation techniques such as horizontal and vertical flips on both seismic and masked images. As a result, we got a total of 1000 different synthetic seismic images. Due to this, the loss was significantly reduced, and the mean IOU was also achieved high as compared to results with 200 images.

2.3. Data Preprocessing

In data preprocessing, we imported seismic images and resized them to 128 * 128 pixels. We did it to better fit the algorithms and would take less time to train the segmentation model. Then we are storing all the resized images into a list since the image

dataset that we are working on is an unstructured dataset and we are converting it into a structured dataset, to manipulate the data.

Now to label each pixel of the masked image, we performed a threshold operation to set the value of white color (255) as 1 and black color (0) as 0. In threshold operation, we set the threshold value as 127. So, pixel value above 127 will be set as 255 and below as 0. In a way, we will have two unique pixel values in the masked image. Further, we divided the image array by 255 to get the label masked image with labels 0 and 1. Class 0 represents the black color and class 1 represents the white color. In the later step, we converted the list into a NumPy array for fast processing.

2.4. Feature Engineering

In feature engineering, we are performing encoding on the images in the binary form where two classes are taken depicting black and white. Label encoding is done in which 1 label is given to images having faults and 0 labels are given to the images which are free from any faults.

3. Methodology

3.1. Methods Description

Feature Extractors

- **ResNet**

ResNet is a type of Artificial Neural Network. It makes use of jumps to skip over some of the layers. Some ResNet models are executed with two-fold or three-fold jumps that comprise nonlinearities like ReLu and batch normalization. Apart from this, an extra weight matrix can be utilized to gain proficiency with jumping the weights. ResNet utilizes a layer of 34 plain network architecture in which the jumps are added later. The jumps transform the architecture into a Residual network. This network can be made using the TensorFlow and the Keras API. At the end of each convolutional layer in ResNet, a batch normalization layer is added. ResNet utilizes four different modules which are all built using the residual blocks. Every module makes use of many residual blocks with the quantity of output channels being the same for all of them.

- **InceptionNet**

InceptionNet is a deep neural network comprising of the components which are repeated known as the Inception modules. The naive Inception module is made up of 5 key components: the Input layer, 1x1 convolution layer, 3x3 convolution layer, 5x5 convolution layer, Max pooling layer, and the concatenation layer. The full Inception module consists of a basic convolution, pooling, and functions to normalize. The full InceptionNet body comprises of 9 such inception modules that are stacked over each other. It consists of a sum of 22 layers in total and approximately 5 million features(parameters). This was the very first version of the InceptionNet called Inception-v1. Inception v-1 had a false rate of just 6.67%.

- **VGG16**

VGG 16 is a convolution neural network model. There are around 16-19 weight layers and around 138 trainable parameters in VGG16. In this architecture, the input image is taken as a constant-sized image of size 224x224. The RGB values are normalized for each pixel as a part of the preprocessing by calculating the difference between the value of each pixel and the average value of all the pixels. The image is made to go through the first stack of 3x3 convolution layers after which ReLu activation is applied. The activations are then made to go through a max-pooling layer which reduced the size of the activations to half. These activations are again made to go through a similar stack. After this, they are passed through a stack of 3 convolutional layers and a max pool layer after which they are again passed through two stacks of 3 convolutional layers followed by the output layer.

Segmentation Techniques

- **Unet**

Unet makes use of the technique of fully convolutional networks for the segmentation process of the images. Unet uses consecutive contrasting layers followed by upsampling operators. The input image is made to go through the model after which 2 convolutional functions with ReLu are applied. This is followed by an encoder and a decoder block. The encoder block reduces the dimensions of the image by using 2 stridden max-pooling layers. Apart from this, there are repeated convolutional layers whose quantity of filters keeps on increasing. The decoder layer consists of convolutional layers with decreasing filter quantity and unsampling takes place at a mild pace till the top layer. The decoder block also uses jumps to connect the past outputs to its layers.

- **FPN**

The FPN takes a single-scale picture as input and results in relatively measured feature maps at numerous levels in a completely convolutional

manner. As a result, it acts as a general solution for constructing feature pyramids which are used in numerous tasks. The pyramid is made in two ways: a bottom-up and a top-down pathway. The bottom-up pathway is the feedforward calculation backbone ConvNet that calculates a hierarchy comprising of feature maps with a scaling step of 2. The previous layer output serves as a reference for the feature maps. The top-down pathway depicts high-resolution features by upsampling coarsely, but strong feature maps from higher levels of the pyramid. The features are upgraded with those of the bottom-up pathway.

- **Linknet**

Linknet is a deep neural network architecture capable of segmenting images that can be utilized for multiple tasks. It attempts to efficiently share the learnings of the encoder with the decoder at the end of every downsampling block. The first block consists of a 7x7 convolution layer after which a 2x2 window-sized max pool layer is applied. The last block does the full convolution after which 2D convolution is applied. In the encoder and decoder block, at the end of each convolution layer batch-normalization and ReLu nonlinearities are applied.

Workflow

The first part of the workflow of our project is that of Data Acquisition. In this part, we collected and extracted the images from the original dataset. After this, the next part was Data-preprocessing. In this part, we read and resized the images and stored them in a list to create a structured dataset. Next, we performed Data Augmentation by performing horizontal flip and vertical flip on the images and thereby, increasing the size and the variance in our dataset and also, reducing the problem of overfitting. The next step in our workflow was Feature Engineering. In this step, we performed labeling on the images giving label 0 to fault-free images and label 1 to the faulty ones. The following step was to split the dataset into testing and training data. We used a split of 85% training data and 15% test data. This was followed by the Model Training and Evaluation step. In this step, we used the Adam optimizer, softmax activation and the CrossEntropy function to calculate the loss. We made use of the Intersection over Union (IoU) metric to evaluate our model. The typical workflow of our project is depicted in **figure 1**.

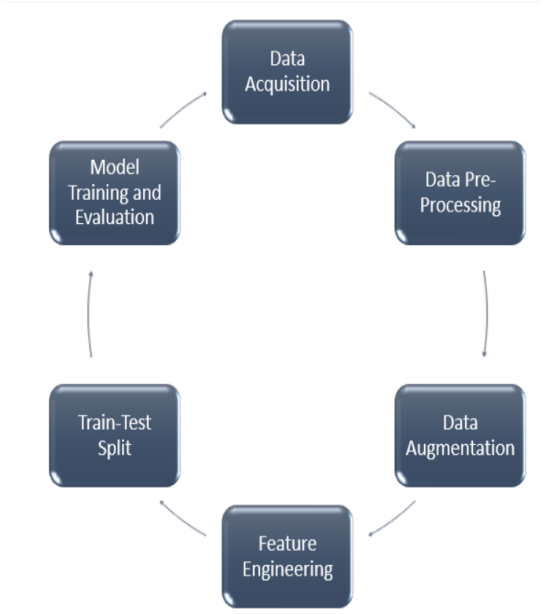

Figure 1-Workflow of our project

Model Training

There are two parts to the training process: preprocessing and CNN model training. We choose N seismic sections with apparent faults from raw inputs as the CNN's training set in this step. We've decided not to create a validation set in this case. The well-trained CNN model is utilized in the prediction section to anticipate defects in other sections.

Preprocessing – For data processing, especially in deep learning, preprocessing is crucial. We Rescaled and resized it to 128x128.

Fault Labeling – Performed thresholding on the value of each pixel – 0-255 where 127 was the threshold. Eventually, the final labels were 0 and 1.

CNN model training – The data was divided 80:20. (800 images in the training set and 200 images in the test set). Now we trained the dataset on each segmentation approach, using all three feature extractions for each technique. There was a separate pre-processing component for each segmentation approach. On the Google Colab GPU, training each strategy took fifteen to twenty minutes, which was a short computation time.\

Parameters – Learning rate as 0.0001, Epochs as 50 and, Batch size as 10.

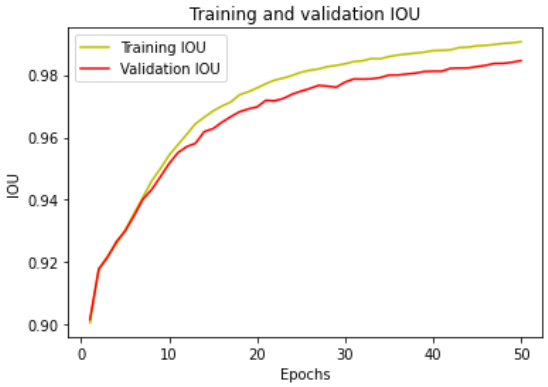

Figure 2-IoU score of FPN with inceptionnet

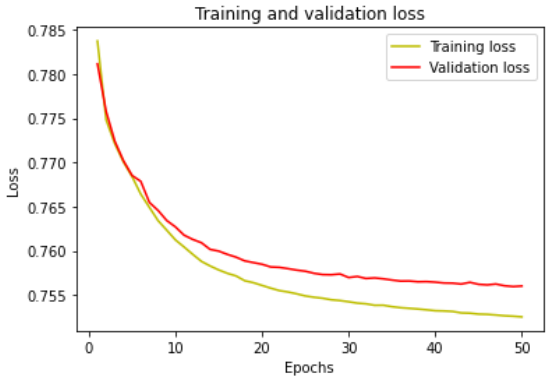

Figure 3-Loss of FPN with inceptionnet

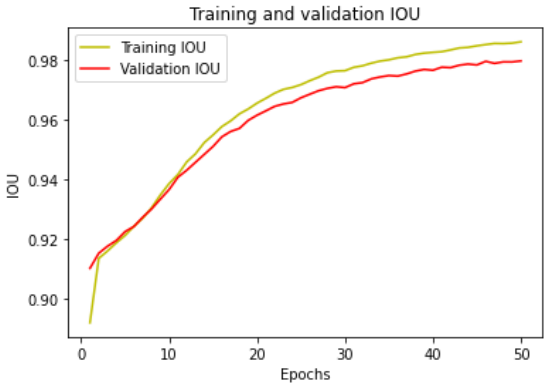

Figure 4-IoU score of FPN with resnet

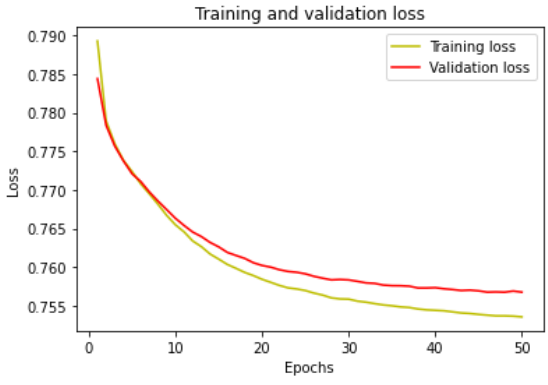

Figure 5-Loss of FPN with resnet

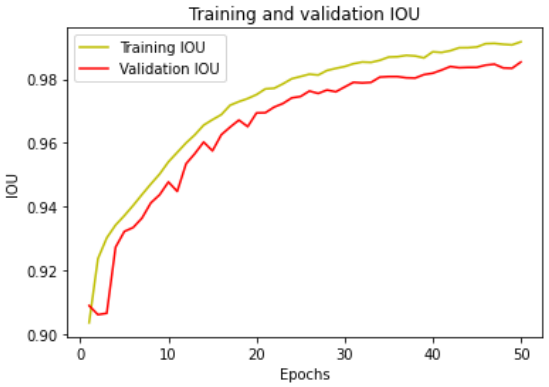

Figure 6-IoU score of FPN with VGG16

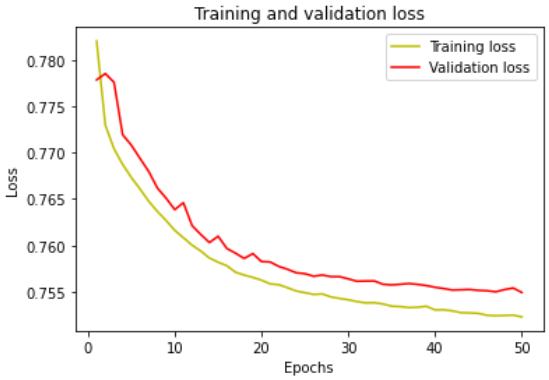

Figure 7-Loss of FPN with VGG16

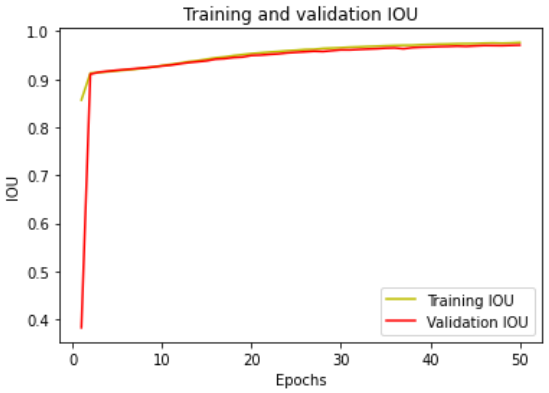

Figure 8-IoU score of Link net with inception net

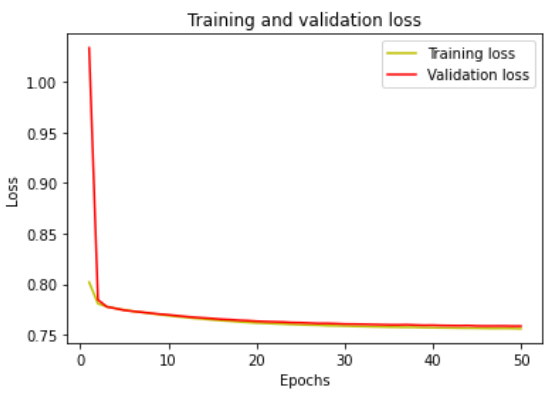

Figure 9-Loss of Link net with inception net

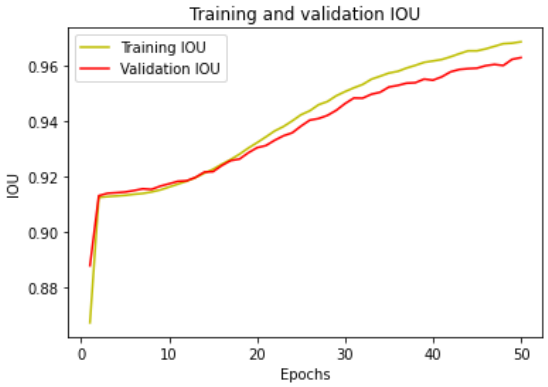

Figure 10-IoU score of Link net with resnet

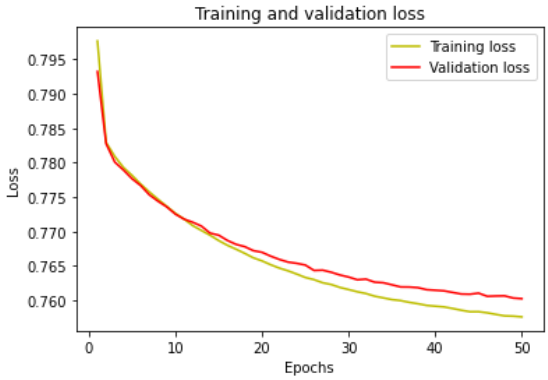

Figure 11-Loss of Link net with resnet

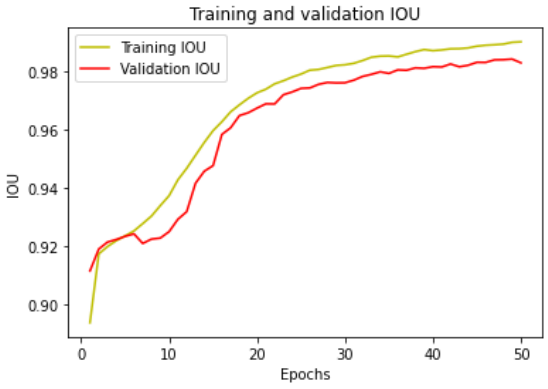

Figure 12-IoU score of Link net with VGG16

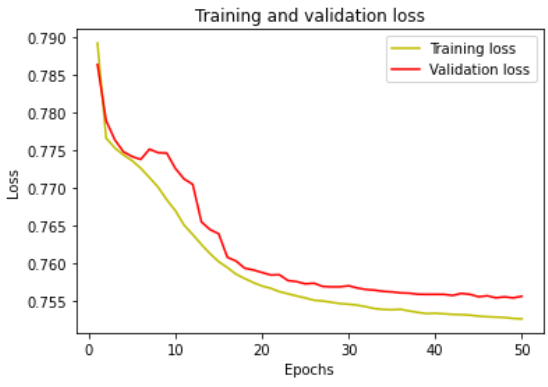

Figure 13-Loss of Link net with VGG16

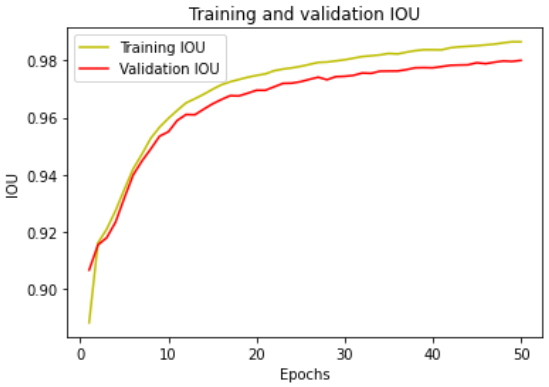

Figure 14-IoU score of Unet with inception net

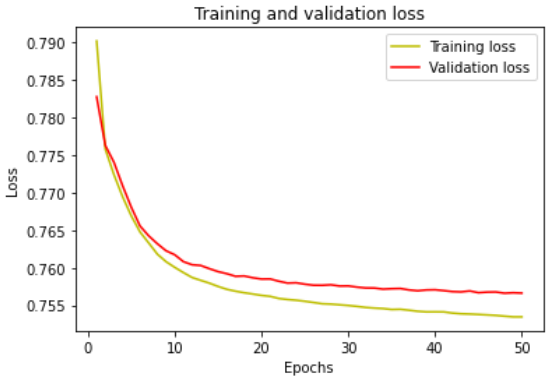

Figure 15-Loss of Unet with inception net

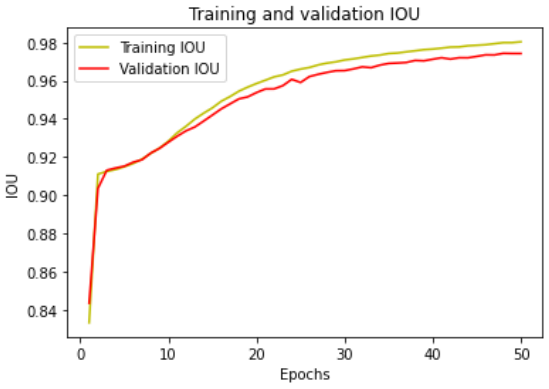

Figure 16-IoU score of Unet with resnet

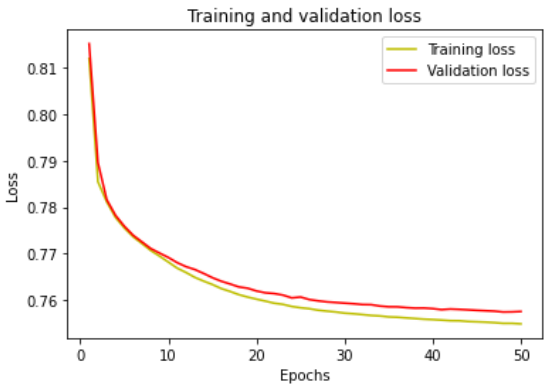

Figure 17-Loss of Unet with resnet

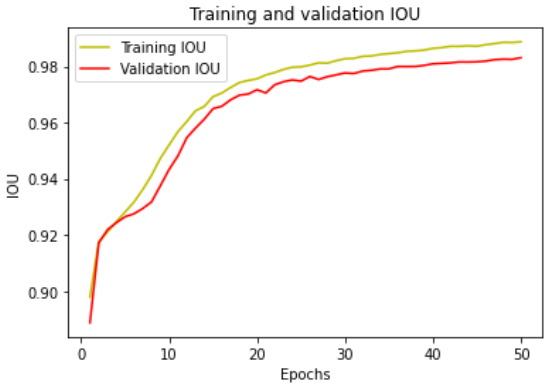

Figure 18-IoU score of Unet with VGG16

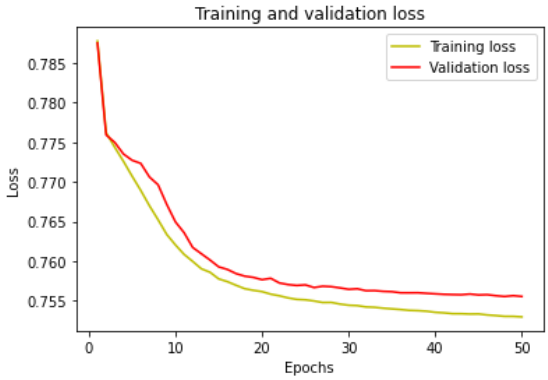

Figure 19-Loss of Unet with VGG16

Model Testing

We tested the final trained models on the 200 images and the output received were masked images. We then further verified the results using the training and testing loss graphs. For each segmentation technique we carried out 50 epochs, so we contained more accurate images for future comparisons.

Results

Intersection over Union (IoU) Metric

The IoU metric is utilized to compute the accuracy of an object detector on a particular dataset. It is actually a technique to give numerical value in the percent of the overlap between the target mask and the predicted output. This metric calculates the ratio of the number of pixels that are similar between the target and the prediction masks and the total number of pixels present in both masks. The IoU score is computed for every class separately. It is then meaned over all classes to give a global, mean IoU score.

The results of the three segmentation techniques along with the VGG16 as a feature extractor on a test image are shown in the figures below.

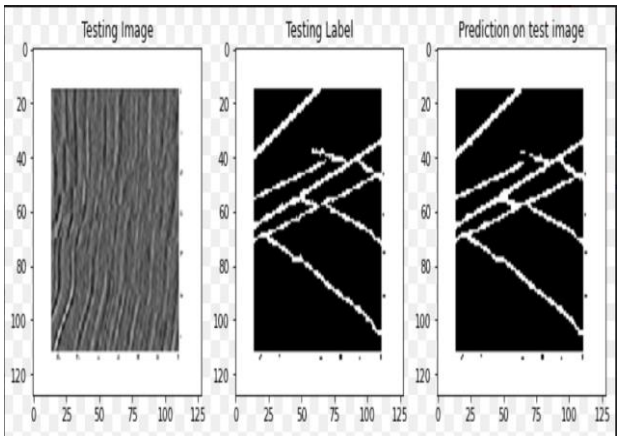

Figure 20-FPN with VGG16(1)

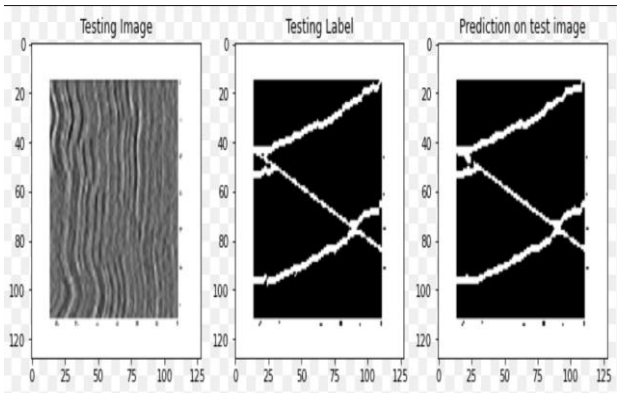

Figure 21-FPN with VGG16(2)

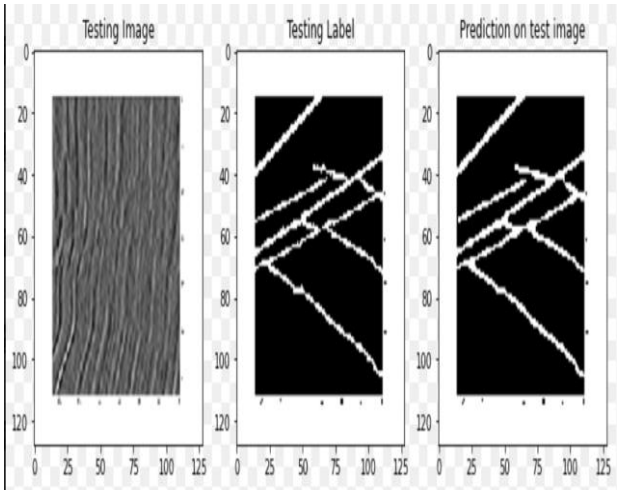

Figure 22-Linknet with VGG16(1)

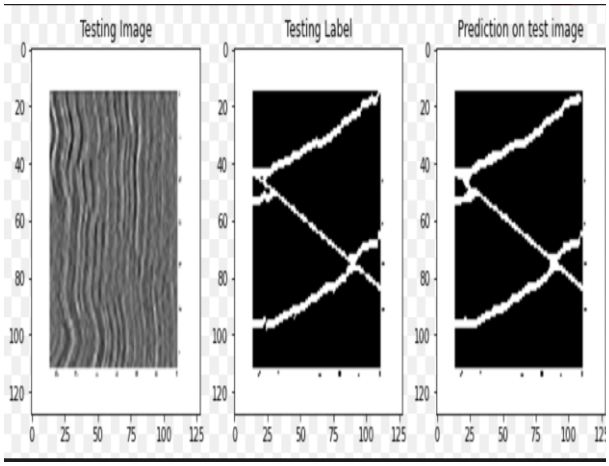

Figure 23-Linknet with VGG16(2)

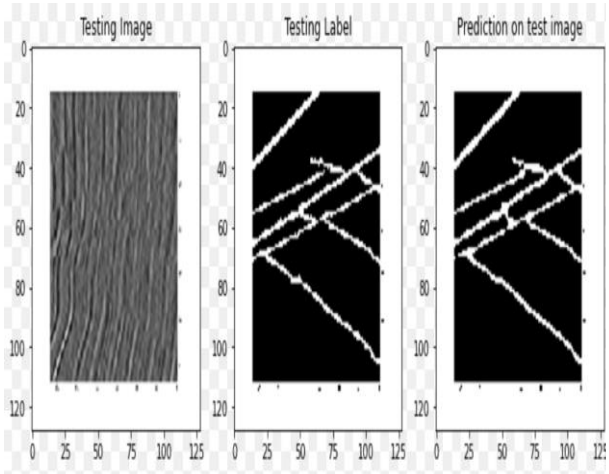

Figure 24-Unet with VGG16(1)

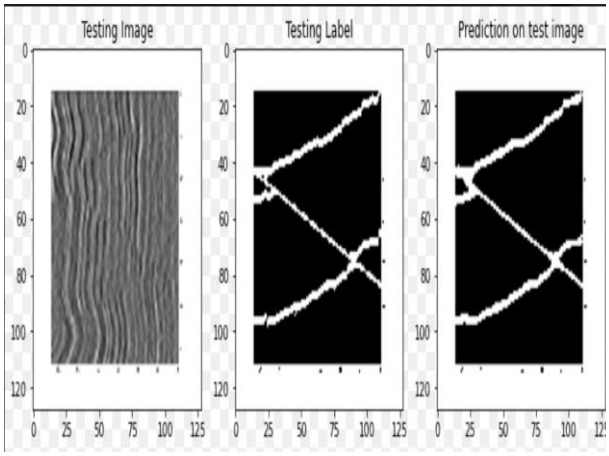

Figure 25-Unet with VGG16(2)

The comparative analysis of the three segmentation techniques and the three feature extractors is shown in the following **Tables 1,2 and 3**.

Link Net		
Feature Extractor	Train	Test
Resnet	0.97	0.962
Inception Net	0.978	0.971
Vgg16	**0.989**	**0.983**

Table 1-Linknet with the three feature extractors

U-Net		
Feature Extractor	Train	Test
Resnet	0.9815	0.954
Inception Net	0.987	0.98
Vgg16	**0.989**	**0.983**

Table 2-Unet with the three feature extractors

FPN		
Feature Extractor	Train	Test
Resnet	**0.987**	**0.979**
Inception Net	**0.9918**	**0.984**
Vgg16	**0.9918**	**0.985**

Table 3-FPN with the three feature extractors

Conclusion

The FPN is the best image segmentation technique among all the three as it has the best train and test mean Intersection over the Union score for all the three feature extractors. VGG16 is the best extractor for all the three image segmentation techniques as it has the highest mean Intersection over Union score for all the image segmentation techniques for both the train and test data.

Future Work

In the future, we will also try with other data augmentation techniques such as adding noise to the images, making the images sharper, etc. Moreover, we will use also perform other image segmentation techniques along with different feature extractors and perform a comparative analysis.

References

[1] Di, H., Shafiq, M., Wang, Z. et al. 2019. Improving seismic fault detection by super-attribute-based classification. *Interpretation* **7** (3): 251-. http://dx.doi.org/10.1190/int-2018-0188.1.

[2] Araya-Polo, M., Dahlke, T., Frogner, C. et al. 2017. Automated fault detection without seismic processing. *The Leading Edge* **36** (3): 208-214. http://dx.doi.org/10.1190/tle36030208.1.

[3] Xiong, W., Ji, X., Ma, Y. et al. 2018. Seismic fault detection with convolutional neural network. *GEOPHYSICS* **83** (5): 97-. http://dx.doi.org/10.1190/geo2017-0666.1.

[4] A. Pochet, P. H. B. Diniz, H. Lopes and M. Gattass, "Seismic Fault Detection Using Convolutional Neural Networks Trained on Synthetic Poststacked Amplitude Maps," in *IEEE Geoscience and Remote Sensing Letters*, vol. 16, no. 3, pp. 352-356, March 2019, doi: 10.1109/LGRS.2018.2875836.

[5] Li, Shengrong, Changchun Yang, Hui Sun and Hao Zhang. "Seismic fault detection using an encoder–decoder convolutional neural network with a small training set." *Journal of Geophysics and Engineering* (2019): n. pag.

[6] Zheng, Zhi Hong, et al. "Multi-Attributes and Neural Network-Based Fault Detection in 3D Seismic Interpretation." Advanced Materials Research, vol. 838–841, Trans Tech Publications, Ltd., Nov. 2013, pp. 1497–1502. Crossref, doi:10.4028/www.scientific.net/amr.838-841.1497.

[7] 210 - Multiclass U-Net using VGG, ResNet, and Inception as backbones - YouTube

[8] RMiftakhov/faultSeg: Using synthetic datasets to train an end-to-end CNN for 3D fault segmentation (We are working on an improved version!) (github.com)