Filling gaps, replacing bad data zones and super-sampling of 3D seismic volumes through Machine Learning

Introduction

Deep Learning algorithms are exceptionally good in repairing images with partly missing data. In image inpainting the missing data is a patch with hard zeros. The algorithm learns to infill the missing data by training it on thousands of examples with known inputs (images with randomly blanked patches) and corresponding outputs (complete images).

Image inpainting can solve a range of similar problems regularly encountered in seismic data. For example: we can use inpainting to enhance or replace bad data zones, or to interpolate missing traces. Similarly, we can halve the bin-size by adding seismic traces in both the inline and crossline directions. Note that this is not a substitute allowing for acquiring at a larger bin-spacing; we are not adding data, we are merely interpolating. However, super-sampling can prove very useful when merging data sets of different vintages.

Image inpainting is typically done on 2D matrices with Red, Green and Blue components as inputs. In this paper, seismic inpainting (hereafter seismic interpolation), is done either on 2D matrices with seismic amplitude inputs, or on 3D matrices with seismic amplitudes.

U-Net Architecture

A U-Net is a Convolutional type of Neural Network that is known as an auto-encoder (Ronneberger et.al., 2015). It consists of decoder part and an encoder part (Fig. 1). The encoder decomposes the input image sequentially into smaller-size features. The decoder recombines the features sequentially into larger-size components until the target image emerges. In other words, a U-Net transforms an image to another image of exactly the same size. This process is known as image-to-image transformation.

U-Nets can be used to solve segmentation problems and regression problems. In the case of segmentation, the target output is a binary mask, e.g., an image (cubelet) with 0’s and 1’s as in fault imaging, or values with 1, 2, …, N as in seismic facies segmentation.

In regression problems, as is the case in seismic interpolation problems, the target images have continuous values.

Figure 1 U-Net architecture. The input (2D image, or 3D cubelet) is decomposed into smaller features in the encoder part (left side). It is then rebuilt into another image (cubelet) by the decoder part (right side). The horizontal lines are skip connections to ensure that no information is lost in the process. Image source: Computer Vision Group, Uni-Freiburg.
Data and Methodology

The test data is a 3D seismic volume covering the city of Delft in The Netherlands. The volume consists of 451 in-lines and 469 cross-lines covering an area of 85 km². The bin-size is 20x20 m. The time range is 0 to 3.8 s with a sampling rate of 4 ms.

In one of the tests, we show the generic applicability of the trained model by applying it to the F3 dataset, which exhibits quite different seismic character. F3 is located approx. 350 km to the North of Delft. The bin-size is 25x25 m, the time range 0 to 1.848 s at 4 ms sampling.

We use OpendTect - Machine Learning platform to: 1) create training sets, 2) train U-Nets, and 3) to apply the trained models. In some tests the training examples are partly overlapping 2D images of 128x128 samples extracted along in-lines. In other tests we extract partly overlapping 3D cubelets of 128x128x64 samples. In all tests the target is the original seismic data. The input images / cubelets are the corresponding examples extracted from modified (partly blanked) 3D volumes.

Example 1: interpolation of missing traces

In this example we train a 3D U-Net (128x128x64 samples) to interpolate missing traces. The input volume is created by randomly blanking approx. 33% of all seismic traces. We extract 3D cubelets from one side of the volume and we test on the part reserved for blind tests. Fig. 2 shows a seismic line from the blind test area. On the left we see the input data with 33% of the traces blanked. In the middle the U-Net prediction result is shown and, on the right, we display the truth, i.e., the original seismic data.

Example 2: generic applicability

In the following example we take the 3D U-Net that was trained on data from Delft, and apply it to F3, a 3D seismic dataset that is located some 350 km to the North. Whereas Delft is an onshore dataset, F3 is located offshore. The geology in both areas is quite different. Also, the seismic acquisition differs (bin-sizes are respectively 20x20 m in Delft and 25 x 25 m in F3). We have no information about the seismic processing but it is fair to state that the seismic character in both data sets is very different. Nevertheless, we apply the trained U-Net model on F3 with 33% randomly blanked traces “AS IS”. We notice that the U-Net result is scaled differently than the original data. Also, some vertical stripes are visible after application of the U-Net interpolator. Both problems are easily solved by applying a whole-
trace RMS scaling to the U-Net result. Fig. 3 shows part of an in-line in F3 with randomly blanked traces, the RMS scaled U-Net reconstruction, and the ground truth.

**Figure 3** Seismic in-line 600 from F3. The input seismic (left) contains 33% randomly blanked traces. The section in the middle is the data that is reconstructed by a 3D U-Net (128x128x64 samples) that was trained on data from Delft. The original data (ground truth) is displayed on the right.

**Example 3: Super-sampling**

Next, we show an example of creating a data set with half the bin-size. The bin-size in Delft is 20x20 m. Our goal is to create a data set with 10x10 m bin-size. Again, we start by creating an input volume from which to extract training examples. We do this by blanking every other trace in both directions. In other words, 75% of all traces in our input volume are blanked. Next, we try to reconstruct the original data with a 3D U-Net of 128x128x64 samples. After testing the trained model on a blind test area, we create a new input dataset. This time, we insert a blank trace after each trace in both directions. This results in another dataset with 75% blank traces but now the traces are spaced 10 m apart in both directions. Application of the trained U-Net results in the desired super-sampled dataset. Again, we stress that this is not a substitute for acquiring less data by doubling the acquisition bin-size. In this process we do not add real information, we merely interpolate data, albeit with a very powerful interpolator. The real benefit of creating a super-sampled dataset in this way is when it facilitates merging of vintage datasets. Fig. 4 shows the input data with every other trace blanked and the U-Net super-sampled result.

**Figure 4** Seismic in-line 2725 from Delft. The input seismic (left) has blank traces inserted at each real trace position in both in-line and cross-line directions meaning 75% of the input traces are blank traces. The super-sampled result displayed on the right is generated by a 3D U-Net (128x128x64 samples).
Example 4: replacing a bad data zone

The Delft dataset features a bad data zone right above the historical city centre. Apparently, no seismic acquisition was permitted in this area. The undershoot area is approx. 35x45 traces wide and 400 ms deep. We cut out the bad data and replace it with a Machine Learning interpolation result. The model is a 2D U-net of 128x128 samples. It is trained on data examples extracted from good data zones between 0 to 508 ms (corresponding to 128 samples). In each of the input examples, we introduce a gap with a randomly chosen width at a random position. The width of the gap varies between 5 and 50 traces. To compensate for the very low amplitudes in the upper section, we scale the examples with a dynamic gain function. In the application phase, we apply a constant scalar to the output. Fig. 5 shows inline 2725 before and after replacing the bad data zone with the U-Net interpolated result.

![Delft City Centre, approx. area affected by undershooting](image)

**Figure 5**  Seismic in-line 2725 from Delft. The bad data zone over the city centre (left) is replaced by interpolated data (right). The interpolator is a 2D U-Net of 128x128 samples along in-lines.

Conclusions

We have shown that a U-Net can be trained to: fill in missing traces, to replace bad data zones, and to create super-sampled datasets. Some trained models are generic interpolators that can be applied without re-training to solve similar issues on completely different datasets at distant locations.

Acknowledgments

We thank our colleagues Arnaud Huck, Hardeep Jaglan and Mark Crawford for their support and encouragement.

References