POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

# Salt Segmentation of Geophysical Images through Explainable CNNs

Laurea Magistrale in Telecommunication Engineering - Ingegneria delle Telecomunicazioni

**Author:** Francesco Maffezzoli

**Advisor:** Prof. Paolo Bestagini

**Co-advisors:** Dr. Vincenzo Lipari, Dr. Francesco Picetti

**Academic year:** 2021-2022

## 1. Introduction

With the increasing need of hydrocarbons worldwide, oil and gas industries are collecting several hundreds of gigabytes of seismic data to find possible oil and gas reservoirs. Up to now, the collected data needed to be processed by field domain experts, such as geologists and geophysicists, to find possible locations of underground hydrocarbons. This hand-made process of analyzing huge amount of data is extremely costly and time consuming for gas industries, but that is not the only issue: it is known that gas and oil reservoirs are usually located near underground salt basins, which are notoriously difficult to image correctly due to their elastic properties.
The task of picking salt bodies in seismic images can be considered a segmentation problem, and can be efficiently solved by deep learning algorithm efficiently like Convolutional neural networks (CNNs). CNNs are widely used in automatic image processing and can be very efficient in terms of training and testing time. From these premises, in this work we focus on:

- Develop a salt segmentation method for seismic images investigating the impact of several loss functions to train a CNN.
- Exploit eXplainable AI (XAI) techniques to better understand which are the leading motivations that make a CNN classify a certain image region as salt.

Our first contribution will help domain experts in their salt analysis work. The second contribution will help in increasing the level of trustworthiness in CNN results.

## 2. State of the Art

This section reports some useful state of the art on salt segmentation and XAI.

### 2.1. State of the art for salt segmentation

Salt segmentation is composed of three different salt picking phases. After each one of them, the produced salt mask is provided to the migration algorithm that adjust the velocity model of migrated image taking into consideration the salt perimeter, thus enabling the next picking phase iteratively. CNNs for salt segmentation can achieve human level accuracy in some specific situations, but greatly suffer dataset domain shifting, meaning that to effectively use CNNs for salt segmentation, a portion of the data set must first be interpreted by humans to create an ad hoc training set for fine tuning. To tackle the problem of domain shift a full learning pipeline called SaltNet [3] have been proposed. SaltNet copes with the absence of unlabeled data from the new dataset by generating proxy labels for CNN training using semisupervised learning: an ensemble of five pre-trained CNNs with different architectures is used to generate an initial prediction on the new, unlabeled dataset. Using CNNs ensembles usually provides better performances than single a single CNN, in this case different architectures have been used to provide different views on the target dataset.

### 2.2. XAI state of the art

The reason why Deep learning is so widely used is that it does not require direct coding of hand-crafted featurs. Trained algorithm can be considered "black boxes" since developers do not need to explicitly code the the solution of a problem but need to pay more attention to the definition of the training procedure. The field of XAI studies methods and techniques to shed light on those black boxes, making them more reasonable and interpretable. Multiple methods are available available to inspect the working principles of CNNs.
As an example, Class Activation Mapping (CAM) was the first XAI technique to introduce the concept of saliency maps: a saliency map is a heat map built over the original image that highlights which part of the image contributed the most to the final prediction. CAM algorithm uses the property of the Global Average Pooling (GAP) layer to address which feature map influences the final decision the most. The GAP layer projects every channel of the CNN latent representation to a dense layer by performing an average on each channel. The saliency map is obtained by simply weighting the feature maps $f_k(x, y)$ with their weight value $w_k$ as

$$C(x,y) = \sum_k w_k f_k(x,y) \tag{1}$$

The obtained weighted sum is a linear combination of all the feature maps and can be upsampled to match the input image shape thus highlighting regions that impact on the classification result.

## 3. Salt Segmentation

This section is devoted to our proposed salt segmentation method.

### 3.1. Problem definition

Salt recognition task can be considered a binary segmentation problem: performing binary segmentation on an image means assigning a binary label to each pixel depending on his content. In the case of salt segmentation, we can define our two labels as **background** and **salt**. To be more specific, given two labels $(S, B)$ representing class salt and background, performing segmentation on in image $I$ means assigning to each of his pixel $I(x, y)$ one of the two labels producing a binary mask $M$ with the same dimension of $I$. Formally, we can define the ideal segmentation mask as

$$M(x,y) = \begin{cases} 0, & \text{if } I(x,y) \in B \\ 1, & \text{if } I(x,y) \in S \end{cases} \tag{2}$$

where $I(x, y) \in B$ means that the pixel in position $(x, y)$ belongs to class $B$ (with a slight abuse of notation).
Our goal is to develop a CNN that is able to produce a mask $M$ as close as possible to the ground truth mask when fed with an input image $I$.

### 3.2. Proposed Salt Segmentation Method

The proposed segmentation method follows the pipeline depicted in Figure 1.
The first step of input preprocessing is patch extraction; 2D images are in fact too large to be used directly for training. The patch extraction implemented is tunable, depending on the needs. One can decide how big a patch should be and how much close patches overlaps (patch shape and stride). The second part of preprocessing is the normalization step. Normalization improves CNNs learning capabilities by standardizing the input to fixed range values, in this case z-score normalization have been applied: given an image patch we simply subtracted its mean and divided by its standard deviation. After preprocessing the training and validation step is performed, which provides a trained CNN ready for testing. At test time, the trained CNN is fed with testing patches and is required to produce binary masks for each of them: once all the

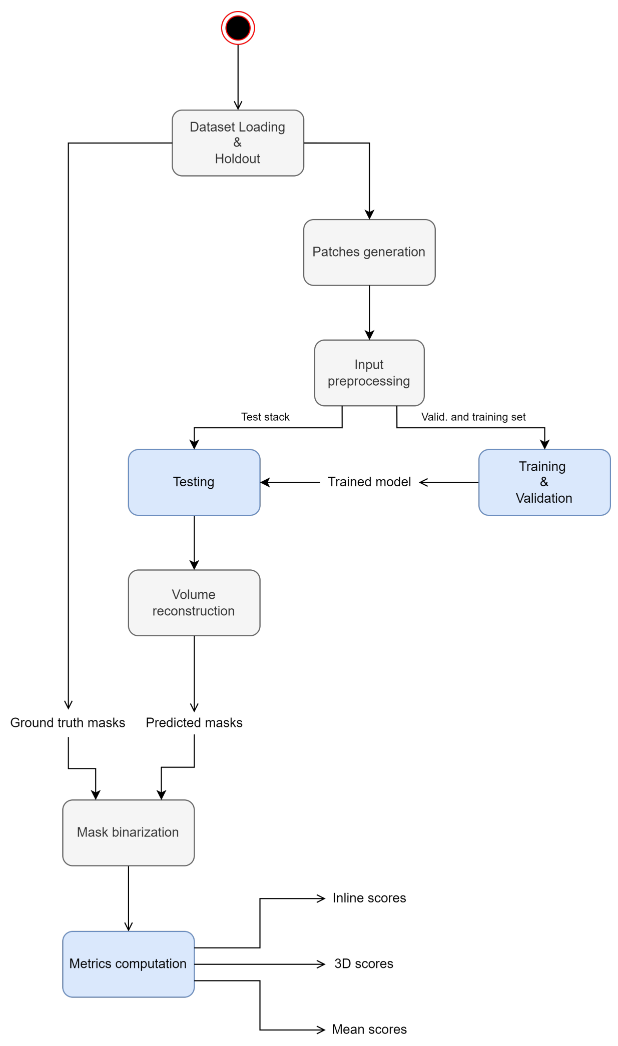

Figure 1: Segmentation process pipeline. Blocks coloured in blue are the core part of the segmentation process.

patches have been processed, those are stitched together forming a complete 2D image. The last step is metric evaluation, the process of comparing ground truth masks and CNN produced masks to compute performance scores for the tested CNN.

## 3.3. Segmentation CNN

The input patch, after being normalized, is fed to the CNN input layer and processed by passing through an encoder and decoder: the chosen network backbone is Unet. Unet architecture is widely known to be very effective for segmentation tasks. Unet architecture is made by an encoder and decoder part: the encoder contracts the input image reducing its dimensionality to a small latent representation, while the decoder up-samples the compressed data to match the input image dimension, creating a mask. On the encoder and decoder architectural side, the selected model is ResNet34.

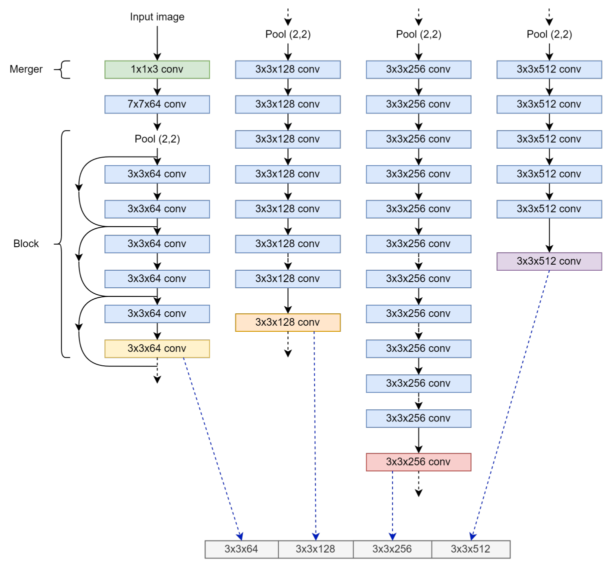

Figure 2: ResNet34 encoder architecture. For simplicity skip connection have been added only on the first block. Coloured layers at the end of each blocks are concatenated to create the final latent representation.

The reasons for this decision are mainly two: ResNet34 has already proven to be optimal for salt segmentation problems in the literature [3], also the same CNN architecture has been selected by our reference baseline, ensuring a faithful comparison of results. ResNet34 is made of four convolutional blocks connected together, at the end of each block, the number of channels doubles, and a pooling layer halves the image dimensionality. In order to use ResNet34 with single-channel images like ours, we added a convolutional layer called "merger" made of three $1x1$ convolutions. Every block of ResNet architecture outputs a stack of channels with different dimensionality: The final latent representation, gray in Figure 2, is formed by concatenating the final output of all four blocks, encapsulating information in four different processing stages.

The main focus of the proposed study on salt segmentation is loss testing. Many different losses are available for segmentation tasks, each of one leads the optimization process to different minima. For the sake of brevity, only the names of the losses will be reported, without introducing any formula or graph associated to them. The selected losses are: Binary Crossentropy (BCE), Contour, Dice, Focal, Hausdorff, Jaccard , Kullback-Leibler (KL) divergence, Lovasz, Mean square error (MSE), and Wing.

## 4. CNN Interpretation

This section will dive into the details of two different XAI techniques we used: Activation maximization (AM) and Network inversion (NI). Each of them have a different scope and tries to inspect how a CNN works under a different prospective.

### 4.1. Activation Maximization

The AM technique is based on the following idea: to understand the meaning of a specific neuron in a CNN, we have to find out which is the input image $x^*$ that maximizes the activation of that neuron. The resulting image could be a good representation of what that neuron is doing. Given a neuron of a neural network with parameters $\theta$, the input image $x^*$ we are looking for can be found by solving the problem [1]

$$x^* = \arg\max_x h(\theta, x), \tag{3}$$

where $h$ computes the neuron output when fed with image $x$. Since CNNs are differentiable with respect to their input, it is possible to find the target input image by tuning it iteratively. The steps to be taken are:

- Initialize an image $x_0$ as random noise for all pixel values.
- Compute the gradient on the input image using back-propagation.
- Update each pixel of the image to maximize the neuron activation by moving on the previously computed gradient: the update iteration is known as gradient ascent.

A shortcoming of the standard AM method is that in this way it is possible to inspect only one neuron at a time. Looking at only one neuron activation might be enough when inspecting a specific class neuron, but is not enough to capture how intermediate layers work. Without introducing any transformation robustness or parameterization, the optimal input image that activates a neuron might be covered with random noise. This happens because the target neuron strongly respond to this high-frequency patterns that covers the image. Transformation robustness substantially increase the quality of target images without penalizing legitimate high frequencies components. The intuition behind transformation robustness is that, since the CNN is usually trained to identify targets at different scales and positions, if an image maximizes activation of a neuron also a slightly transformed version of it does.

### 4.2. Network Inversion

NI can be used to effectively inspect a layer as a whole, both convolutional or fully connected [2]. To do so a dataset sample image $x$ is fed into the CNN, and his representation is computed up to the layer $A$ to be inspected obtaining the so called "feature map" $A(x)$. The algorithm goal is to find an input image $x^*$ whose feature maps $A(x^*)$ is close to $A(x)$, adding also a regularization therm such that $x^*$ resembles in some ways the dataset sample $x$.
The optimal image $x^*$ is obtained iteratively by optimizing an input image $x$ such that [2]

$$x^* = \arg\min_x(||A(x) - A(x^*)||^2 - \lambda(x)) \tag{4}$$

The algorithm tweak iteratively the input noise $x_0$ until the minimum distance between feature maps $A(x_0)$ and $A(x)$ is reached. The main goal of NI is to find out what part of input image information is lost and what is kept at a specific network layer.

## 5. Results

In this thesis two datasets have been used in total. One, the SEAM dataset, has first been used to train CNNs for segmentation purposes. SEAM dataset has then been processed to perform classification on segmentation networks for the purpose of CNN explainability. To transform the segmentation dataset in a classification one, a label has been assigned to each patch depending on the percentage of salt present. The second dataset used is called LANDMASS, a classification dataset that contains images belonging to four different types of seismic artifacts: Reflectors, Chaotic horizons, Faults and Salt domes.

| Loss | DICE | PS |
|---|---|---|
| BCE | 0.8914 | 0.748591 |
| Contour | 0.902134 | 0.773261 |
| Dice | 0.923534 | 0.88268 |
| Focal | 0.890961 | 0.727423 |
| Hausdorff | 0.893505 | 0.936658 |
| Jaccard | 0.917728 | 0.784634 |
| KL | 0.907488 | 0.798635 |
| Lovasz | 0.907314 | 0.903268 |
| MSE | 0.824938 | 0.692863 |
| Wing | 0.90502 | 0.923319 |
| BCE + Dice | 0.905626 | 0.862574 |

Table 1: Summary of 3D scores obtained from segmentation experiments with different losses

## 5.1. Segmentation results

To evaluate segmentation CNN performance many score functions have been used, for the sake of brevity we will report here only Dice and Perimetric similarity (PS). PS have been selected since we are mainly interested in delineating the border of salt domes: PS is in fact Volumetric score only on the Salt perimeter. Table 1 reports scores obtained by testing the proposed losses. The last raw of the table represent the "baseline", that is a CNN trained with the same Unet-Resnet34 architecture but with combined loss function. The baseline loss is a linear combination of BCE and Dice coefficient (DIC) loss functions, with weights equal to one. Baseline details have been given to us by ENI; the purpose of this testing session is to see if some other, less common losses can perform better. Dice loss scored the best Dice score among all the tested losses, meaning that Dice loss performs overall better with respect to the other loss functions. Hausdorff on the other hand reached the top PS, meaning that Hausdorff is the best loss for salt perimeter recognition.

## 5.2. Interpretation results

CNN Inversion techniques discussed in Section 4 investigate on how a CNN interprets different classes: to apply AM for example, we need some specific target to optimize that embodies how our CNN sees one class or another. It is impossible to apply those techniques directly to a segmentation networks since it maps 2D inputs in 2D outputs, there are no specific objective that resembles one class or another. To cope with this problem we come up with two different alternatives:

- Remove the decoder from a pre-trained segmentation CNN, append a classification head and train the new CNN lowering the encoder Learning rate (LR).
- Train a new classification CNN from scratch.

Only in the first case we are effectively inspecting a CNN trained for segmentation purposes by keeping the encoder almost the same. For brevity purposes, only SEAM dataset images produced on the pre-trained segmentation network are reported, leaving aside LANDMASS dataset CNNs.
The CNN to be inspected uses a pre-trained encoder for segmentation purposes, classification head is then trained on a new classification task splitting salt patches with a threshold of 20%. Note that there is no gap between background and salt class, so definition of salt and background can be ambiguous at times.

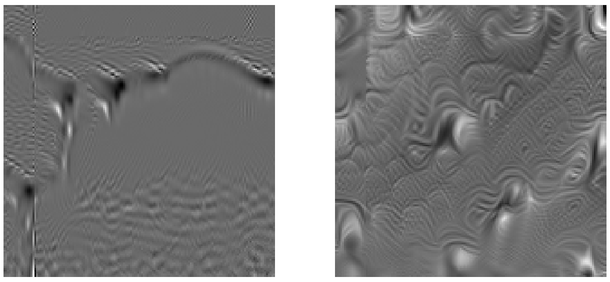
(a) AM on class salt on segmentation CNN (b) AM on class salt on classification CNN

Figure 3: AM on class salt considering the segmentation (a) and classification (b) CNN.

Figure 3a shows AM applied to class neuron of the segmentation CNN. Note how salt body discontinuity is sharp and stands out with respect to the rest of the image: this tells that the CNN has effectively learned to recognize salt bodies by looking at surface discontinuity between background and salt. The same AM algorithm on classification CNN (Figure 3b) suggests that the CNN only learned to recognize the general discontinuity between salt and background. Figure 4 shows NI applied to the segmentation CNN. Note how the deeper layer we inspect, the more the input image is modified by the encoder.

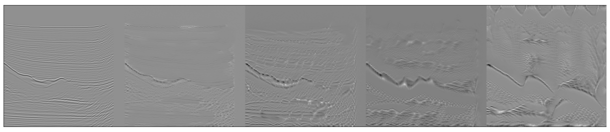

Figure 4: NI applied to a pretrained CNN. The firs image on the left is the input image.

Figure 4 shows a segmentation CNN hardly modify the input image, leaving intact all the details: this happens probably because segmentation CNNs need to retain as much spatial information as possible about the input image to be able to reconstruct the output mask with great precision. CNNs trained appositely for classification do not need to "understand" salt bodies in such a detailed manner since classification task does not need to retain any spatial information about the input image.

## 6. Conclusions

In this work we tackled the problem of salt segmentation in two different ways: in the first part we proposed a baseline segmentation CNN and took care of all the steps, from data preparation to testing and metric evaluation. In the second part we inspected network behavior using two XAI techniques called Network Inversion and Activation Maximization. Now that all the considered losses have been successfully tested, the next step to make to reach even higher segmentation performances is hyperparameter tuning, one further improvement to reach state of the art performances could be implementing ensemble-based predictions using multiple architectures. On the side of CNN inspection, we successfully implemented AM and NI techniques to ResNet34 architecture achieving highly interpretable results. By looking ad AM-produced images, we explained how a segmentation encoder interprets class salt and background: the produced images contain usually only one very detailed replica of the selected class, embodying the concept of salt or background label. NI technique confirmed the made hypotheses by showing how differently the two encoders process the input image: the segmentation encoder barely modify the image shape and size, classification encoders modify the input dramatically already from from shallower layers, progressively loosing image details leaving behind only its general concept. One bigger step could be implementing a generator network using Generative adversarial network (GAN) training: AM generated images can be used as preferred initialization to train a generator network using GAN training, resulting in impressively realistic images. By reducing the number of optimization steps of the already implemented AM algorithm one can produce perfect starting images for generator networks.

## References

[1] Dumitru Erhan, Y. Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *Technical Report, Univeristé de Montréal*, 01 2009.

[2] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5188–5196, 2015.

[3] Satyakee Sen, Sribharath Kainkaryam, Cen Ong, and Arvind Sharma. Saltnet: A production-scale deep learning pipeline for automated salt model building. *The Leading Edge*, 39:195–203, 03 2020.