Article

Segmentation of Oil Spills on Side-Looking Airborne Radar imagery with Autoencoders §

Antonio-Javier Gallego [1,3]*, Pablo Gil [2,3], Antonio Pertusa [1,3], and Robert B. Fisher [4]

1   Pattern Recognition and Artificial Intelligence Group, Department of Software and Computing Systems, University of Alicante; {jgallego, pertusa}@dlsi.ua.es
2   Automation, Robotics and Computer Vision Group, Department of Physics, Systems Engineering and Signal Theory, University of Alicante; pablo.gil@ua.es
3   Computer Science Research Institute, University of Alicante, Alicante, Spain
4   School of Informatics, University of Edinburgh, Edinburgh, UK; rbf@inf.ed.ac.uk
*   Correspondence: antonio.gallego@ua.es; Tel.: +349-65-903772

1   **Abstract:**   In this work, we use deep neural autoencoders to segment oil spills from Side-Looking
2   Airborne Radar (SLAR) imagery. Synthetic Aperture Radar (SAR) has been much exploited for
3   ocean surface monitoring, especially for oil pollution detection, but few approaches in the literature
4   use SLAR. Our sensor consists of two SAR antennas mounted on an aircraft, enabling a quicker
5   response than satellite sensors for emergency services when an oil spill occurs. Experiments
6   on TERMA radar were carried out to detect oil spills on Spanish coasts using deep selectional
7   autoencoders and RED-nets. Different configurations of these networks were extensively evaluated
8   and the best topology significantly outperformed previous approaches, correctly detecting 100%
9   of the spills and obtaining an F_1 score of 93.01% at the pixel level. The proposed autoencoders
10 perform accurately in SLAR imagery that has artifacts and noise caused by the aircraft maneuvers,
11 in different weather conditions and with the presence of look-alikes due to natural phenomena such
12 as shoals of fish and seaweed.

1. Introduction

16   A quick response from governments is required in situations of marine pollution due to oil
17 spills [1]. When an oil slick is detected, the authorities activate the emergency protocols in order
18 to control the environmental impact and the ecological damage in the sea. The most relevant
19 technologies and spaceborne sensors for oil-spill sensing are described in [2] and [3,4]. CleanSeaNet
20 is an example of a monitoring service of oil spills and vessels provided by the European Maritime
21 Safety Agency (EMSA). Governments use mainly two kinds of sensors to carry out the monitoring
22 of the sea surface: Synthetic Aperture Radar (SAR) installed on satellites (ERS-1/2, JERS-1, Envisat
23 ASAR, RADARSAT-1, RADARSAT-2, COSMO-SkyMed, Sentinel-1, Sentinel-2, ALOS-2, TerraSAR-X
24 among others) as in CleanSeaNet, and Side-Looking Airborne Radar (SLAR) or another airborne
25 miniaturized radar as in [5]. Both sensors can be used for oil slick detection.
26   The SLAR used in this work is a SAR mounted on aircraft instead of a satellite and it has two
27 radar antennas. SLAR and SAR sensors have some differences as mentioned in [6]. On the one
28 hand, SLAR has a range and resolution smaller than SAR and, consequently, the complexity in the
29 detection is higher due to the lower details in the acquired image. However, SLAR does not depend

on the orbit because it is mounted on an aircraft, and therefore it has a better response time than SAR. As aircraft can modify their altitude and flight path during signal acquisition, SLAR images have different perspective and scale. In addition, these images have artifacts and noisy areas caused by the aircraft motion (turns, slips, etc.) and by the location of the two SLAR antennas under the aircraft wings. These artifacts and types of noise are not present in SAR images in which speckle (with granular appearance) is the most common noise.

The oil-spill detection strategies using SAR can be categorized into two groups. The first contains all the approaches that use the raw signals of the radar as well as polarimetric (PolSAR) or interferometric features (InSAR), and so forth to discriminate the oil slicks [7,8]. The second includes the methods that use intensity images as a representation of the backscattering coefficient of the signal [9,10]. In addition, some works such as [11] combine image and polarimetric features extracted from oil spills and look-alikes in order to discriminate between both targets.

In the state of the art there are many works which address oil spill detection using machine learning techniques. These methods include Tree Forests [12], Support Vector Machines (SVM) [13,14], Generalized Linear Models (GLM) and Boosting trees among others, as in [15,16] where both a Bayesian classifier and several evolutionary algorithms were used to select image features for classifying oil spills and look-alikes. Neural networks have also been used for this task, using as input different features from radar images characterizing an candidate oil slick [17,18]. The choice of the classifier architecture is dependent on the problem and when the features are properly selected there are no significant differences in the results, as shown in [19] with PolSAR data.

In some previous works, image processing and computer vision algorithms were used to automatically extract features and segment regions from radar images. These data can be fed to a network such as in [20], in which two neural networks were proposed, one to detect dark formations and another to classify them as oil slicks or look-alikes. In the past, neural network architectures typically had only three layers (input, hidden and output) as in [21], where a Multilayer Perceptron (MLP) and the Radial Basis Function (RBF) networks were used. The classification can be performed at two levels of detail: classification of pixels representing oil slicks when the number of images in the dataset is small but they have a high resolution [22] or scenarios where the dataset contains many images [11].

More recently, many approaches based on deep learning techniques have been proposed to increase the success rate in image classification tasks. The main motivation of using deep convolutional neural networks (CNN) is their ability to extract suitable features for the task at hand, as it is very difficult to properly select the features that can allow us to discriminate between oil spills and other natural phenomena due to the similarity of their representations as dark areas in the image. In this line, Chen *et al.* [23] selected and optimized the PolSAR features reducing the feature dimensions used as input of the classifier to distinguish oil spill and biogenic look-alikes through layer-wise unsupervised pre-training. For this task they use Stacked AutoEncoders (SAE, autoencoders with multiple layers) and Deep Belief Networks (DBN). In addition, Guo *et al.* [24] proposed a CNN to identify dark areas in SAR images as crude oil (oil slick), plant oil and oil emulsion (both look-alikes), reaching average success rates of 91% vs the 80% of a traditional neural network. In all these works, authors used SAR imagery.

There are many oil slick detection methods that use SAR imagery as input. However, it is uncommon to find detection methods using SLAR imagery. Two recent works in this line were presented in [6] and [25]. The first one is based on traditional image segmentation techniques whereas the second one uses Recurrent Neural Networks (RNN) to perform the detection. Two decades ago, Ziemke [26] already proposed a RNN using SAR images for oil spill detection, showing robustness to variations in both weather conditions and illumination changes.

Unlike the previous works using SLAR, we propose an approach that is able to detect oil slicks even in the presence of look-alikes. Our method, which is an extension of a previous study presented in [27], is focused on solving the oil-slick region segmentation problem using deep learning

techniques, particularly denoising autoencoders using Convolutional Neural Networks as encoder and decoder functions.

The rest of the paper is structured as follows: Section 2 introduces background on autoencoders, Section 3 presents the proposed method, followed by the dataset description in Section 4, the evaluation in Section 5, and finally the conclusions and future work in Section 6.

2. Autoencoder architecture

Autoencoders were proposed decades ago by Hinton and Zemel [28], and since then they have been an active research field [29]. Autoencoders consist of feed-forward neural networks trained to reconstruct their input, that is, the input and the output must be exactly the same. This problem may seem trivial as their goal is to learn the identity function $f(x) = x$, but in practice we impose some restrictions in order to force it to generate a compressed intermediate representation. This is achieved by using intermediate layers with a size smaller than the inlet layer. This bottleneck forces the network to extract the most representative characteristics of the sample that allow its subsequent reconstruction, thus generating a meaningful intermediate representation.

Figure 1 shows a graphical scheme of an autoencoder. This type of network is divided in two parts, the first part (called the *encoder*) receives the input and creates a latent (or encoded) representation of it, and the second part (the *decoder*) takes this intermediate representation and tries to reconstruct the input. Formally speaking, given an input x, the network must minimize the divergence $L(x, g(f(x)))$, where f and g represent the encoder and decoder functions, respectively. The encoder function provides a meaningful compact representation which might be of great interest as regards feature learning or dimensionality reduction [30].

Some variations of autoencoders have been proposed in the literature to solve other kind of problems. For example, *denoising autoencoders* [31] are an extension trained to reconstruct the input x from a corrupted version (usually generated using Gaussian noise) of it, denoted as $\hat{x}$. Thus, these networks are trained to minimize the divergence $L(x, g(f(\hat{x})))$, therefore they not only focus on copying the input but also on removing the noise [31–33].

Autoencoders, and particularly denoising autoencoders, have been successfully used in many fields such as music, character recognition or medical image segmentation, but in addition they are currently being used in remote sensing to perform recognition and scene classification. For example, Zhao *et al.* [34] combined Stacked Autoencoder (SAE) and Extreme Machine Learning (ELM) techniques for target recognition from raw data of High-Resolution Range Profile (HRRP) acquired from three different aircraft, achieving a faster time response than other deep learning models. Others authors such as Kang *et al.* [35] used 23 baseline features and three-patch Local Binary Pattern (LBP) features that were cascaded and fed into a SAE for recognition of 10-class SAR targets. In addition, Liang *et al.* [36] presented a classification method based on Stacked Denoising Autoencoders (SDAE) in order to classify pixels of scenes (acquired from GF-1 high resolution satellite) into forest, grass, water, crop, mountains, etc.

In this paper we propose to use autoencoders which receive as input the signal of SLAR sensors and return as output the areas detected as oil spills.

3. Proposed method

Based on the idea of *denoising autoencoders*, we use a type of segmentation autoencoder as proposed in [37] but specifically designed for oil spill detection. In this case, we do not aim to learn the identity function as autoencoders do, nor an underlying error as in denoising autoencoders, but rather a codification that maintains only those input pixels that we select as relevant. This is achieved by modifying the training function so that the input is not mapped identically at the output. Instead, we train it with a ground truth of the input image pixels that we want to select. From here on, we will refer to this model as *Selectional AutoEncoder* (SelAE). The SelAE is trained to perform a function
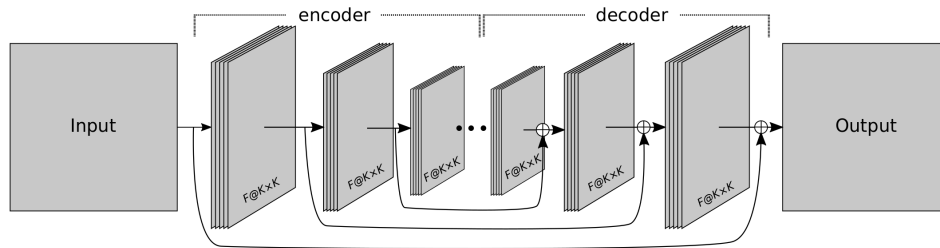
Figure 1. Example of a RED-Net topology. The number of layers can change according to the chosen topology. The symbol $\oplus$ denotes element-wise sum of feature maps. *F* represents the number of selected filters and $(K \times K)$ the size of the kernel.

such that $s : \mathbb{R}^{(w \times h)} \to [0,1]^{(w \times h)}$, or in other words, a binary map over a $w \times h$ image that preserves the input shape and outputs the decision in the range of $[0,1]$.

Following the autoencoder scheme, the network is divided into encoding and decoding stages, where the encoder and decoder functions can be seen as a translator between the input, the intermediate representation, and the desired segmentation. The topology of an SelAE can be quite varied. However, we have considered only convolutional models because they have been applied with great success to many kinds of problems with structured data, such as images, video, or audio, demonstrating a performance that is close (or even superior in some cases) to the human level [38].

The topology of the network consists of a series of convolutional plus *Max Pooling* layers until reaching an intermediate layer in which the encoded representation of the input is attained. It then follows a series of convolutional plus upsampling layers that generates the output image with the same input size. All layers have *Batch Normalization* [39] and *Dropout* [40], and use *ReLU* as activation function [41].

The last layer consists of a set of neurons with sigmoid activations that predict a value in the range of $[0,1]$. Those pixels whose selection value exceeds the *selectional* level δ – which can be seen as a threshold– are considered to belong to an oil spill, whereas the others are discarded.

In addition, in this work we incorporate into this architecture a series of residual connections as proposed in [42]. This type of topology, called RED-Net (Very deep Residual Encoder-Decoder Network), includes residual connections from each encoding layer to its analogous decoding layer (see Figure 1), which facilitates convergence and leads to better results. Moreover, down-sampling is performed by convolutions using stride, instead of resorting to pooling layers. Up-sampling is achieved through transposed convolution layers, which perform the inverse operation to a convolution, to increase rather than decrease the resolution of the output.

We applied a *grid-search* technique [43] in order to find the network architecture with the best configuration of layers and hyperparameters (filters of each convolution, the size of the kernel, and the dropout value). The results of this experimentation are included in Section 5.4, although we anticipate the best topologies for each network in Table 1.
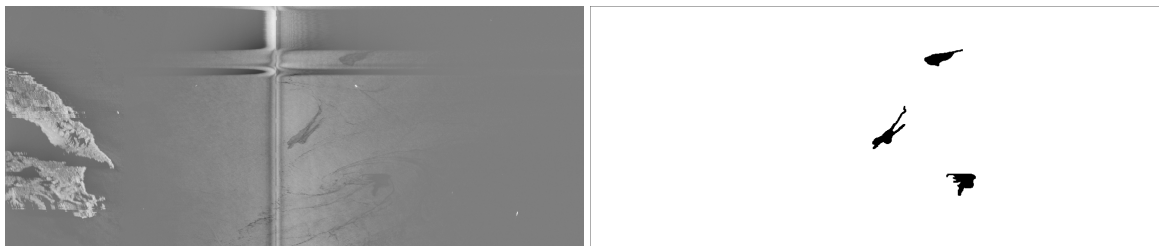
Figure 2. Example of a SLAR sequence from our dataset (left) and its corresponding ground truth (right) with the oil spills labeled at the pixel level. The SLAR image shows an island on the left side, a vertical zone of noise caused by junction of the signal from the two antennas of TERMA radar, and two horizontal bands of noise at the top produced by aircraft maneuvers.

Table 1. Best architectures found after the grid-search process.

Autoencoder type:	SelAE	RED-Net
Input image size:	256x256 px	384x384 px
Number of layers:	4	6
Residual connections:	No	Yes
Filters per layer:	128	128
Kernel size:	5x5	5x5
Down-sampling:	MaxPool(2x2)	Stride(2x2)
Dropout (%):	0	0
***Selectional* threshold δ:**	0.5	0.8

3.1. Training stage

As autoencoders are feed-forward networks, they can be trained by using conventional optimization algorithms such as gradient descent. In this case, the tuning of the network parameters is performed by means of stochastic gradient descent [44] considering the adaptive learning rate proposed by Zeiler [45]. The loss function (usually called *reconstruction loss* in autoencoders) can be defined as the squared error between the ground truth and the generated output. In this case, we use the cross-entropy loss function to perform the optimization of the network weights during a maximum of 100 epochs, with a mini-batch size of 8 samples. The training process is stopped if the loss does not decrease during 10 epochs.

In order to train the network we generated a ground truth marking those pixels of the SLAR input images which correspond to oil spills. Figure 2 shows an example of a SLAR sequence (left) and its corresponding ground truth (right) with the oil spills labeled in black.

In this work, the network is fed with the raw data and the ground truth segmentations, so it must learn to discriminate the areas with oil spills from the rest of the data. That is, no preprocessing is performed on these images to eliminate the noise, as happens in other approaches such as in [46], nor any postprocessing is done to refine the detection.

The next section details all the information about the dataset and the SLAR images used.

4. Dataset

In order to validate the effectiveness of the proposed method, we used a dataset containing 38 flight sequences supplied by the Spanish Maritime Safety and Rescue Agency (SASEMAR). SASEMAR is the public authority responsible for monitoring the Exclusive Economic Zones (EEZ) in Spain and its procedures are based on reports from the European Maritime Safety Agency (EMSA).

176 The data provided by the SLAR sensor of each of these sequences was digitized in images with a
177 resolution of 1,150×481 pixels.

178 The SLAR samples were acquired by a TERMA SLAR-9000 mounted on a variant of the
179 EADS-CASA CN-235-300 aircraft for search-and rescue missions (see Figure 3). This aircraft model
180 reaches a maximum cruise speed of 236 kn, a flight range of 1,565 nmi and around 2,700 nmi with
181 and without payload, respectively. Its flight endurance is close to 9 hours. The SLAR samples are
182 digitalized as 8-bit integers due to the constraints of the monitoring equipment installed on the
183 aircraft. Our autoencoder architecture uses as input these SLAR images in the same format in which
184 they were generated by the TERMA software.

185 The dataset was captured by the aircraft on Spanish coasts at an approximate average altitude
186 of 3,271 feet (although the most common altitude for our missions was around 4,550 feet) and with a
187 wind speed ranging between 0 and 32 kn, the most usual being 14 kn.

Figure 3. SASEMAR 102 (Variant of CN-235-300 aircraft model for search-and-rescue missions) used
to obtain the SLAR sequences, manufactured by EADS-CASA.

188 As stated before, for the ground truth we used a binary mask for each SLAR image, delimiting
189 the pixels corresponding to oil spills. It is important to note that this labeling is performed at a pixel
190 level since the goal is to evaluate both the detection and the precise location of the spills. This way
191 we can provide relevant information such as the position, the size and the shape of oil slicks in order
192 to track them.

193 Figure 4 shows four examples of SLAR images from our dataset. They contain several oil spills
194 (marked with a bounding box in Figures 4a and 4b), along with other types of artifacts such as boats
195 (small bright points), coast (Figure 4d), look-alikes, and noise. Figures 4b and 4c contain many
196 examples of look-alikes, with elongated shapes that are very similar to those of actual spills. All
197 figures show a central band of noise, which is produced by the union of the information from the two
198 SLAR sensors. In addition, the upper part of Figures 4a and 4d shows the noise generated by turning
199 maneuvers of the airplane and the effect produced when the aircraft changes its altitude, respectively.

200 From the 38 flight sequences, 22 contain examples of oil spills and 4 of look-alikes. Within these
201 examples, the spots only represent 0.32% of the pixels in the image, which creates a very unbalanced
202 dataset. To evaluate the method properly in the presence of unbalanced data we use the F_1 and in
203 addition other metrics described in section 5.1.

204 In all the experiments we used an n-fold cross validation (with $n = 5$), which yields a better
205 Monte-Carlo estimate than when solely performing the tests with a single random partition [47]. Our
206 dataset was consequently divided into n mutually exclusive sub-sets, using the data of each flight
207 sequence only in one partition and maintaining the percentage of samples for each class. For each
208 fold, we used one of the partitions for test (20% of the samples) and the rest for training (80%).
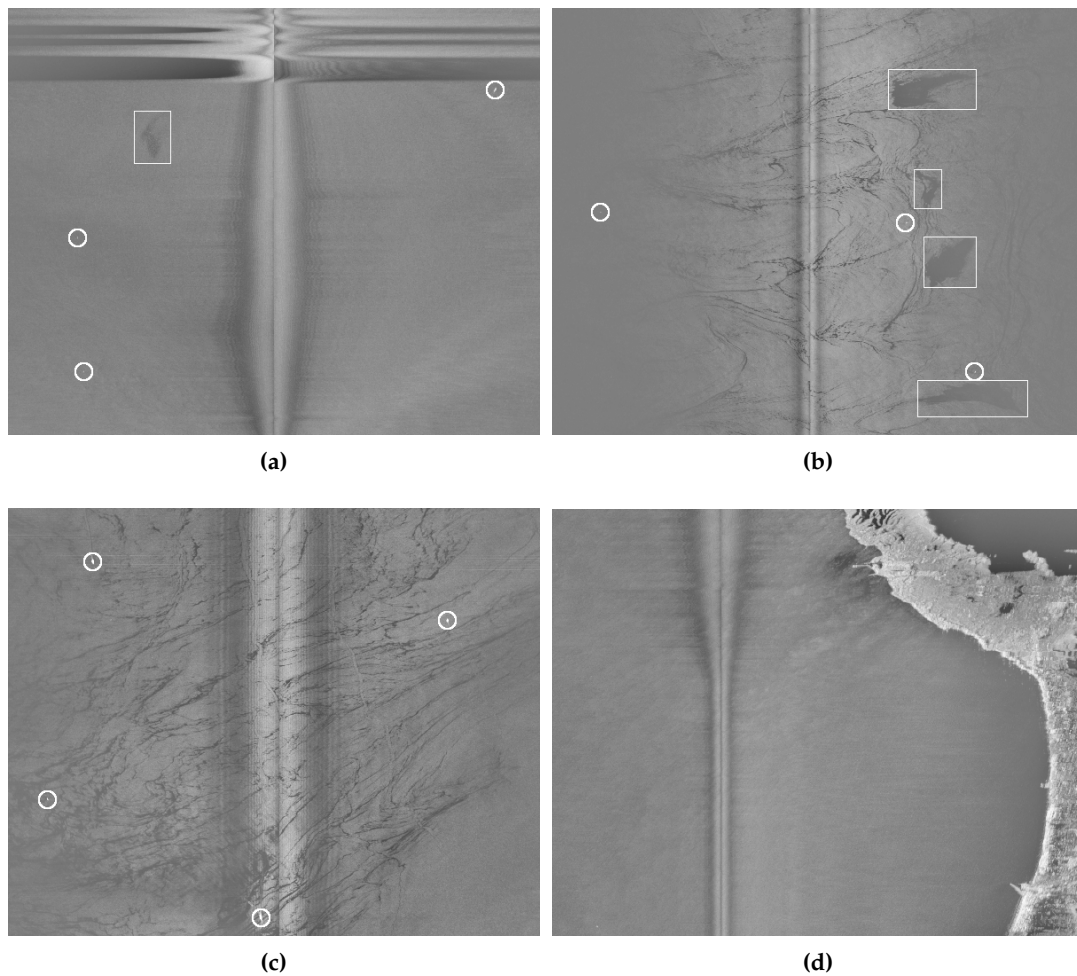
Figure 4. Examples of SLAR images from our dataset showing oil spills (marked with a bounding box), ships (small bright points marked with circles), look-alikes (elongated shapes in figures (b) and (c)), the noise produced by the sensor (the central vertical band that appears in all the images) and the aircraft maneuvers (the horizontal bands that appear in the upper part of figure (a)), and an example of coast (on the right side of figure (d)). Figures (c) and (d) do not contain any example of oil spills, however they have other artifacts that can lead to confusion.

For tuning the hyperparameters (see Section 5.4) the training partition was divided into two, assigning 10% of these samples for validation and the rest (70%) for training. The classifier was trained and evaluated n times using these sets, after which the average results plus the standard deviation σ were reported.

5. Evaluation

This section shows the experiments performed. First, we describe the metrics used for the evaluation, followed by the augmentation methodology and the type of normalization applied. Then we present the best hyperparameters found by the grid-search process and finally, the results obtained by the proposed method.

The following experiments were made on an SGI ICE XA system[1] with two 2.1 GHz, 18-core Intel(R) Xeon E5-2695 (Broadwell) and 256GB RAM. The computational resources from this machine were mainly exploited to parallelize the grid-search process in order to explore several network configurations.

5.1. Evaluation metrics

Three evaluation metrics widely used for this kind of tasks were chosen to evaluate the performance of the proposed method: Precision, Recall, and F_1, which can be defined as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{1}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2}$$

$$F_1 = \frac{2 \cdot \text{TP}}{2 \cdot \text{TP} + \text{FN} + \text{FP}} \tag{3}$$

where TP (True Positives) denotes the number of correctly detected targets (pixels), FN (False Negatives) the number of non-detected or missed targets, and FP (False Positives or false alarms) the number of incorrectly detected targets.

It should be noted that the F_1 metric is suitable for unbalanced datasets but it is not the most adequate for this task since it measures the precision of the results at the pixel level but not whether the algorithm has detected the spill or not. For this reason, we also use the Intersection over Union (IoU) for evaluation, measuring whether the algorithm correctly detects all the spills present in the image and also how well it detects their size and location.

In order to calculate the IoU, we map each object proposal (p) to the ground-truth (g) bounding box with which it has a maximum IoU overlap. Bounding boxes are calculated to include the groups of pixels (or *blobs*) marked as 1 in the network output after applying the selectional threshold or in the ground-truth images. A detection is considered as TP if the area of overlap (a_o) ratio between the predicted bounding box (B_p) and the ground-truth bounding box (B_g) exceeds a certain threshold (λ) according to the following equation:

$$a_o = \frac{area(B_p \cap B_g)}{area(B_p \cup B_g)}$$
$$TP = a_o > \lambda \tag{4}$$

where $area(B_p \cap B_g)$ depicts the intersection between the object proposal and the ground truth bounding box, and $area(B_p \cup B_g)$ depicts its union. By convention, we use a threshold of $\lambda = 0.5$ to set a TP candidate.

5.2. Normalization

Initially we conducted an experiment to determine the best type of normalization for the task at hand. The literature cites different ways to normalize the data to feed a network [48,49], but the most appropriate technique depends on the particular problem. The most common normalization methods are:

---

[1]   Cirrus UK National Tier-2 HPC Service at EPCC (http://www.cirrus.ac.uk).

$$Z_{standard} = \frac{M - mean(M)}{std(M)} \tag{5}$$

$$Z_{min-max} = \frac{M - min(M)}{max(M) - min(M)} \tag{6}$$

$$Z_{mean} = M - mean(M) \tag{7}$$

$$Z_{norm} = M/255 \tag{8}$$

where M is the input matrix containing the raw image pixels from the training set. For the normalization of the test set we used the same mean, deviation, max, and min values calculated for the training set. It is also important to note that the range of values obtained depends on the equation used, however this is not an issue since the configuration of the network allows it, and as stated before, this can lead to a better result.

We evaluated these types of normalization on the two networks, including the option of not normalizing the data. For this we considered a base configuration (with 32 filters per layer, a kernel size of 3×3, a dropout of 0.25, and a *selectional* threshold δ of 0.5), and then we varied the input size (subsampling the input images to 128x128px and 256x256px) and the number of hidden layers of each network (from 4 to 8), in order to obtain an statistically significant average result. The networks were trained using a data augmentation of 20 (see Section 5.3) on the training set, and for the evaluation we used the validation set.

The results of this experiment (in terms of F_1, see Equation 3) are shown in Table 2, where each cell shows the average of 30 experiments (6 network configurations per 5 folds). As can be seen, the best F_1 for the two types of networks are obtained using the standard normalization, followed by the mean norm. The type of data normalization considerably affects the result obtained, since the differences in some cases reach up to 25%. For this reason, in the following experiments we use the standard normalization.

Table 2. Average F_1 (%) plus σ when applying different types of normalization on the input data, and without normalization.

	None	$Z_{Standard}$	$Z_{min-max}$	Z_{mean}	Z_{norm}
SelAE	54.33 ±2.23	**70.02** ±1.26	44.65 ±3.14	69.84 ±1.67	44.10 ±3.57
RED-Net	65.25 ±1.97	**75.12** ±1.07	53.66 ±2.75	74.91 ±1.35	59.67 ±2.91

5.3. Data augmentation

Data augmentation is applied in order to artificially increase the size of the training set [49,50]. As the experimental results show, augmentation systematically improves the accuracy.

To this end, we randomly applied different types of transformations on the original images, including horizontal and vertical flips, zoom (in the range [0.5, 1.5] times the size of the image), rotations (in the range [-10°, 10°]), and shear (between [-0.2°, 0.2°]).

In order to evaluate the improvement obtained with this augmentation process, we carried out an experiment in which we gradually increased the number of random transformations applied to each image from our training set, and evaluated it using the validation set. As before, we performed this experiment for both architectures fixing the configuration to 32 filters per layer, a kernel size of 3×3, a dropout of 0.25, and a *selectional* threshold δ of 0.5, and only varying the input size (subsampling the input images to 128x128px and 256x256px) and the number of hidden layers of each network (from 4 to 8). The input data was normalized using standard normalization.

Figure 5 shows the average results of such experiment, where the horizontal axis indicates the augmentation size and the vertical axis the F_1 obtained. As can be seen, for the two models evaluated

280 the highest improvement is obtained at the beginning, after which the results begin to stabilize and
281 stop improving after 20 augmentations. For this reason, in the following we set to this value the
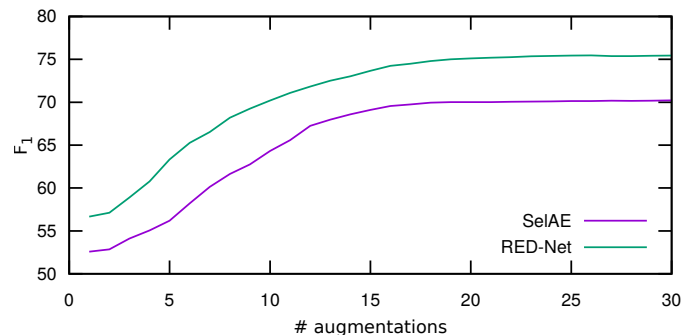282 number of augmentations applied to each image.

Figure 5. Average results of the data augmentation process. The horizontal axis represents the number of augmentations and the vertical axis the average F_1 (in percentage) obtained for each of the networks.

5.4. Hyperparameters evaluation

284 In order to select the best hyperparameters for the two types of CNN evaluated, we have
285 performed a *grid-search* [43] using the training and validation sets. The configurations evaluated
286 include variations in the network input size (from 32px to 512px per side), in the number of layers
287 (from 4 to 8), the number of filters (between 16 and 128), the kernel size (between 3 and 7), the
288 percentage of dropout (from 0 to 0.5), and the *selectional* threshold δ (between 0 and 1). Overall,
289 6,480 experiments were made, using 1,296 configurations $\times$ 5 folds. In all cases we applied an
290 augmentation of 20 and the standard normalization.
291 Figure 6 shows the results of this experiment. The average F_1 when varying the input size is
292 shown in Figure 6a. As can be seen, larger inputs are beneficial for this task. The SelAE architecture
293 obtains the higher F_1 with a 256x256px size, whereas the most suitable size for RED-Net is 384x384px.
294 Figure 6b shows the results when varying the number of layers. The SelAE architecture obtains the
295 best F_1 with four layers, whereas RED-Net requires 6 layers. This may happen because pooling layers
296 lose information for the reconstruction, whereas RED-Net mitigates this loss through residual layers.
297 Figure 6c shows the average F_1 obtained when varying the number of filters per layer. Using more
298 filters increases the F_1, and this improvement is noticeable from 16 until 64 filters, only increasing
299 marginally with 128 filters. Figure 6d shows the average F_1 for the three kernel sizes evaluated, and
300 both architectures obtained the best results with 5x5 filters. Figure 6e shows the average F_1 obtained
301 by varying the dropout percentage applied to each layer. The best result for both architectures in this
302 experiment was obtained without using dropout. The RED-Net results remain stable but they lightly
303 worsen when increasing the dropout, whereas with SelAE the F_1 is significantly lower when dropout
304 grows. Finally, Figure 6f shows the result by varying the *selectional* threshold δ. RED-Net remains
305 fairly stable to changes in this value, obtaining its maximum for a threshold of 0.8. SelAE seems to be
306 more affected by changes, obtaining better results with an intermediate threshold of 0.5.
307 In conclusion, the final architecture chosen for each network is with 128 filters with 5x5 size
308 and without dropout. The SelAE uses an input size of 256x256px, 4 layers, and a threshold of 0.5,
309 whereas RED-Net uses 384x384px with 6 layers and a threshold of 0.8. Table 1 shows a summary of
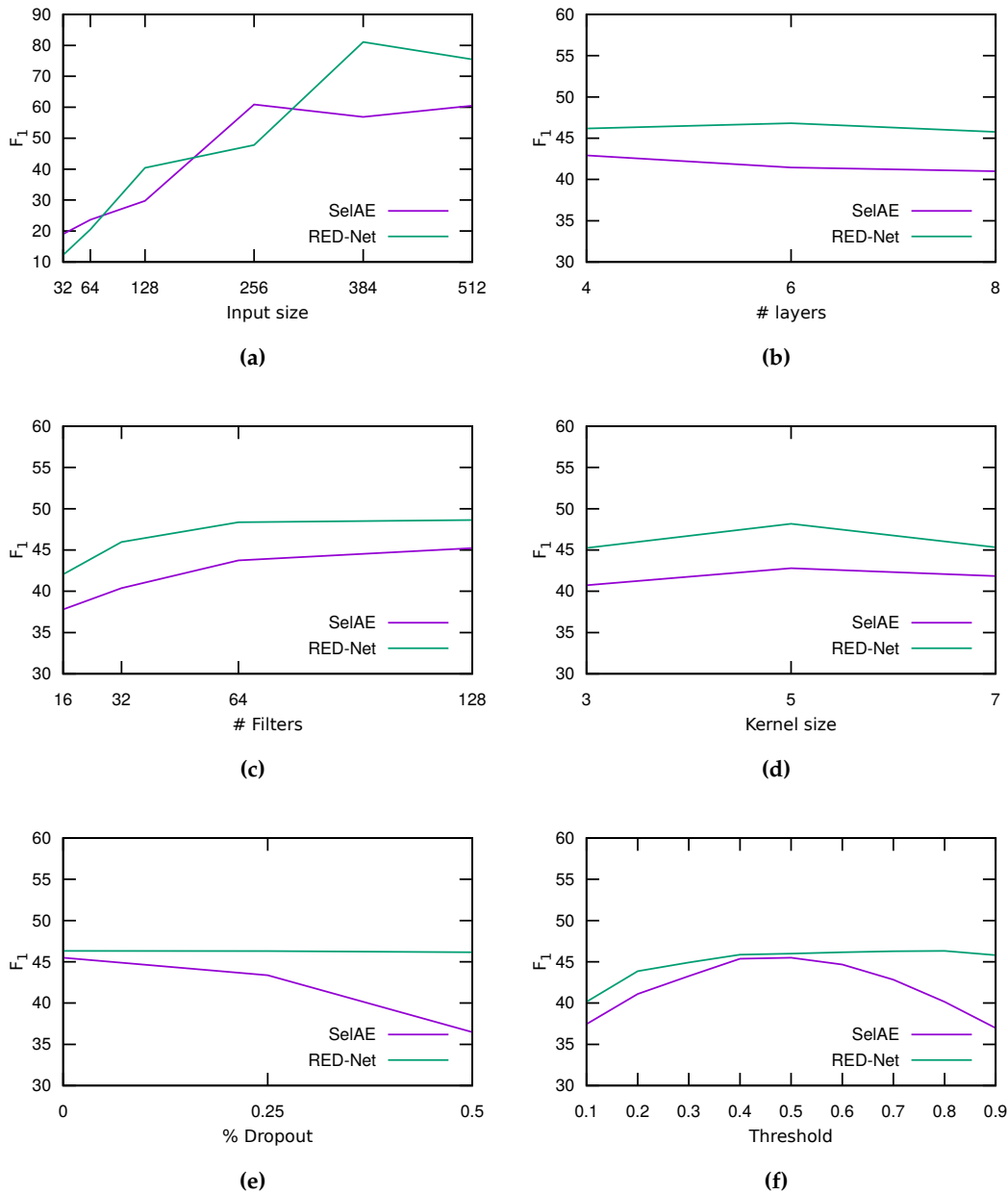310 the topologies that were eventually chosen.

Figure 6. Average F_1 (%) of the grid-search process when varying (a) the input image size, (b) the number of layers, (c) the number of filters per layer, (d) the kernel size of the convolutional filters, (e) the percentage of dropout, and (f) the *selectional* threshold δ.

5.5. Results

Once the best configuration and parameter settings for each network were selected, we evaluated the results using the different metrics. Moreover, we compared these results with three state-of-the-art methods for oil slick segmentation in SLAR images:

- *"Graph-based method"* [51]: It is an adaptation for SLAR images of the method proposed in [52]. It uses progressive intensity gradients for extracting regions with variable intensity distribution.
- *"JSEG method"* [51]: It is also an extension to SLAR images of a previous work [53], where the input image is quantized according to the number of regions to be segmented. Pixel intensities

are replaced by the quantized label building a class-map called J-image. Later, a region-growing technique is used to segment the J-image.

- "*Segmentation guided by saliency maps (SegSM)*" [6]: It first applies a pre-process of the noise caused by aircraft movements using Gabor filters and Hough Transform. Then, the saliency map is computed and used as seeds of a region-growing process that segments the regions that represent oil slicks.

Details regarding the implementation and the parameters used in these methods can be found in the corresponding references.

Table 3 shows the final result obtained with the proposed approach as well as the comparison with the state-of-the-art methods using the test set for the evaluation. It should be noted that the test set had not been used in previous experiments to avoid adjusting the network architecture for this set.

The best results were obtained in all cases using the RED-Net architecture, which shows a higher F_1 (see Equation 3) for all the tested images. On the one hand, the best RED-Net configuration increases up to 3.7 % the F_1 of the SelAE autoencoder, and between a 37 % and a 64 % the other methods of the state of the art. The SegSM method has a high precision and a low recall, which indicates that it accurately detects some parts of the spills but producing many FN. On the other hand, both JSEG and Graph-based methods have a high recall and a low precision, since in this case they are producing many false positives. The proposed method obtains a more balanced result in the detection of oil-spill pixels. This fact can be confirmed by looking at the IoU metric (see Equation 4), where RED-Net also improved significantly the results with respect to the other methods, which indicates a better precision in the detection of the shape and the position of the oil slicks.

Table 3. Evaluation results including the standard deviation for the two architectures using the chosen parameters after grid-search.

Model	Precision	Recall	F_1	IoU
Graph-based	32.99 $\pm$1.62	97.25 $\pm$0.33	48.28 $\pm$1.87	32.55 $\pm$0.16
JSEG	17.04 $\pm$0.32	92.58 $\pm$0.25	28.73 $\pm$0.46	16.50 $\pm$0.35
SegSM	98.54 $\pm$0.27	39.55 $\pm$1.21	55.78 $\pm$1.18	87.33 $\pm$0.51
SelAE	89.64 $\pm$0.95	88.99 $\pm$0.91	89.31 $\pm$0.93	92.14 $\pm$7.21
RED-Net	93.12 $\pm$0.86	92.92 $\pm$0.84	**93.01** $\pm$0.85	100.00 $\pm$0.00

Figure 7 shows a graphic representation of the results obtained with the best approach, i.e. the RED-Net model. The first column of images shows the original input SLAR images (oil spills are marked with a bounding box), and the second column shows the prediction of the network. In the images of the second column, white and black areas depict correct detections of sea and oil spills, respectively, and red and blue pixels depict FP and FN of oil spills, respectively.

These figures help to visualize the accuracy of the proposed model and to understand where the errors of each target class occur. As can be seen, wrong detections are typically made only at the contours of the oil spills.

Figure 7a shows that the proposed method correctly detects the spill even in the presence of noise due to look-alikes (biological origin and weather conditions). In Figure 7c we can see a larger spill produced by a ship emptying its bilge tanks. This spill is correctly detected and there are only few mistakes at the edges. Figure 7e contains coast, but the method do not miss and it correctly detects just one small spill at the center. Figure 7g also shows a coast section in the upper-right part, and in this case the image contains an airplane turn. In this example, even when the spill is located at the center of the noise, the method is able to correctly perform the detection. Finally, the last example in

Figure 7i shows an image with high noise (caused by aircraft movements), including coast at the left side and without any spill. As can be seen, the method correctly concludes that the image does not contain any spill.

6. Conclusions and Future Work

In this work, we propose to use deep convolutional autoencoders for the detection of oil spills from SLAR imagery. Two different network topologies have been analyzed, conducting extensive experiments to get the best type of data normalization, to know the impact of data augmentation on the results, and to obtain the most suitable hyperparameters for both networks.

A dataset with a total of 28 flight sequences was gathered on Spanish coasts using TERMA SLAR radar, labeling the ground-truth in order to train both selectional autoencoders and RED-Nets. It is composed of oil spills acquired in a wide variety of sea conditions dependent on weather (i.e. wind speed) and geographic location as well as of flight conditions such as altitude and type of motion.

The proposed approach is able to segment accurately oil slicks despite the presence of other dark spots such as biogenic look-alikes, low wind which also introduces a lot of look-alikes, and noise due to bad radar measurements caused by the aircraft maneuvers. Results show that the RED-Net achieves an excellent F_1 of 93.01% when evaluating the obtained segmentation at the pixel level. In addition, by analyzing the precision of the regions found using the Intersection over Union (IoU) metric, the proposed method correctly detects the 100% of the oil spills, even in the presence of artifacts and noise caused by the aircraft maneuvers, in different weather conditions and with the presence of look-alikes.

Future work includes increasing the dataset size by adding more labeled samples from additional missions. In addition, Generative Adversarial Networks (GAN) such as Pix2Pix [54] could be used to deal with a reduced dataset by generating synthetic samples. Also, the detected oil slick locations could be used to initialize oil spill models for better oil spill prediction and response [55]. A study correlating the F_1 score with the wind speed or weather conditions could also be useful to understand to what extent the effectiveness of the proposed method depends on these factors.

Author Contributions: A-J.G. conceived the proposed method and performed the experiments; R.B.F. helped with the design of experiment distribution on supercomputer; All authors analyzed the data and results; A-J.G. and P.G. wrote the paper; A.P. and R.B.F. reviewed and helped with clarifying the paper.

Conflicts of Interest: The authors declare no conflict of interest.

Bibliography

1. Liu, Y.; Macfadyen, A.; Ji, Z.G.; Weisberg, R. *Monitoring and Modeling the Deepwater Horizon Oil Spill: A Record-Breaking Enterprise*; American Geophysical Union: Washington, D.C., 2011.

2. Leifer, I.; Lehr, W.J.; Simecek-Beatty, D.; Bradley, E.; Clark, R.; Dennison, P.; Hu, Y.; Matheson, S.; Jones, C.E.; Holt, B.; Reif, M.; Roberts, D.A.; Svejkovsky, J.; Swayze, G.; Wozencraft, J. State of the art satellite and airborne marine oil spill remote sensing: Application to the {BP} Deepwater Horizon oil spill. *Remote Sensing of Environment* **2012**, *124*, 185 – 209.

3. Fingas, M.; Brown, C. Review of oil spill remote sensing. *Marine Pollution Bulletin* **2014**, *83*, 9–23.

4. Fingas, M.; Brown, C. A review of oil spill remote sensing. *Sensors* **2018**, *18*, 91:1–18.

5. Jones, C.; Minchew, B.and Holt, B.; Hensley, S., Studies of the Deepwater Horizon Oil Spill With the UAVSAR Radar. In *Monitoring and Modeling the Deepwater Horizon Oil Spill: A Record Breaking Enterprise*; American Geophysical Union: Washington, D.C., 2011; Vol. 195, pp. 33–50.

6. Gil, P.; Alacid, B. Oil Spill Detection in Terma-Side-Looking Airborne Radar Images Using Image Features and Region Segmentation. *Sensors* **2018**, *18*, 151:1–15.

(a) SLAR input image

(b) Detected oil spill

(c) SLAR input image

(d) Detected oil spill

(e) SLAR input image

(f) Detected oil spill

(g) SLAR input image

(h) Detected oil spill

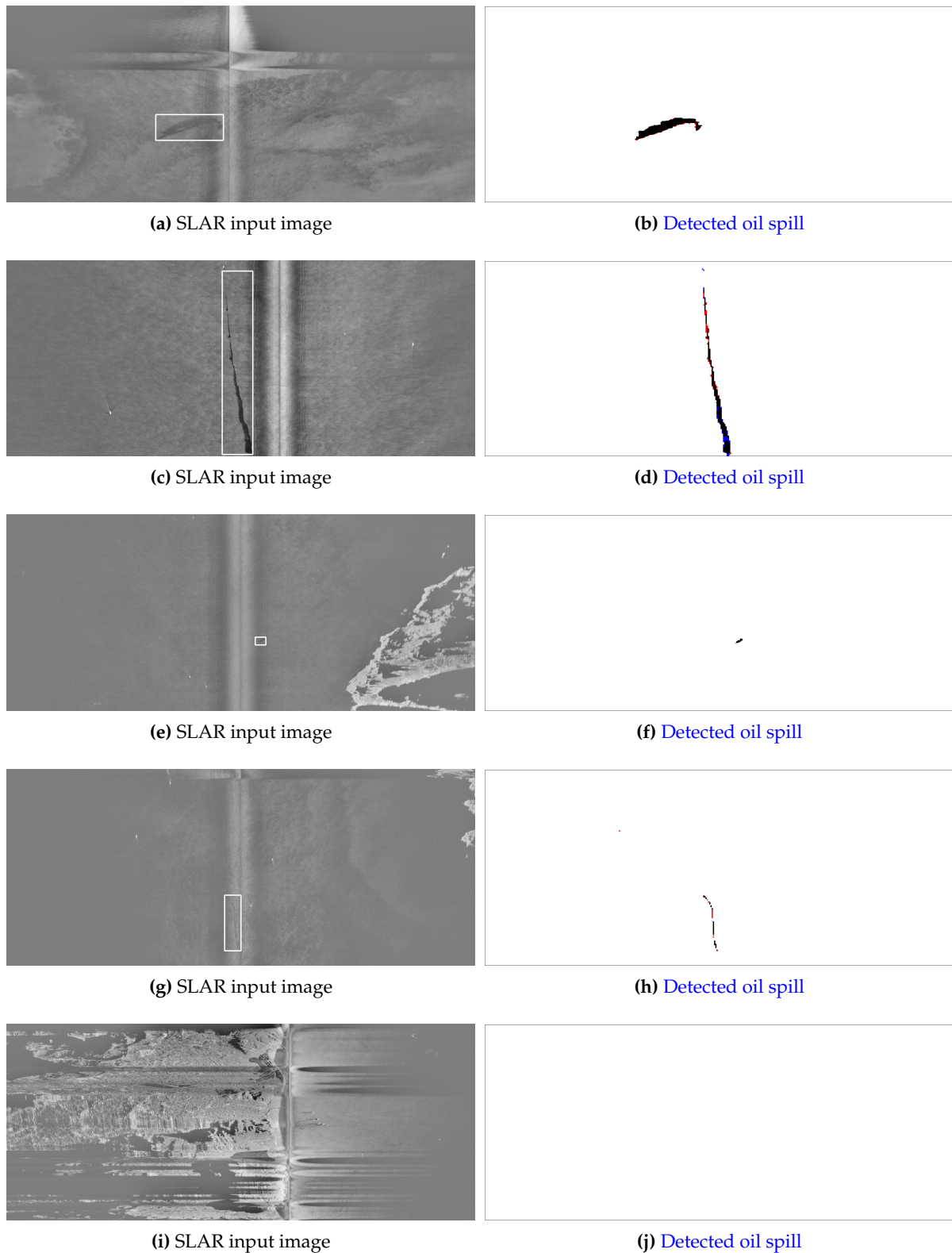**(i)** SLAR input image

(j) Detected oil spill

Figure 7. Results of processing five SLAR input images. The first column shows the original SLAR images (oil spills are marked with a bounding box), and the second column shows the detection results. White and black areas depict correct detections of sea and oil spills, respectively, and red and blue pixels (hard to see because they are few) depict FP and FN of oil spills, respectively.

7. Skrunes, S.; Brekke, C.; Eltoft, T. Characterization of Marine Surface Slicks by Radarsat-2 Multipolarization Features. *IEEE Transactions on Geoscience and Remote Sensing* **2014**, *52*, 5302–5319.

8. Salberg, A.; Rudjord, O.; Solberg, H. Oil spill detection in hybrid-polarimetric SAR images. *IEEE Transactions on Geoscience and Remote Sensing* **2014**, *52*, 6521—-6533.

9. Brekke, C.; Solberg, A.H.S. Classifiers and Confidence Estimation for Oil Spill Detection in ENVISAT ASAR Images. *IEEE Geoscience and Remote Sensing Letters* **2008**, *5*, 65–69.

10. Wu, Y.; He, C.; Liu, Y.; Su, M. A backscattering-suppression-based variational level-set method for segmentation of SAR oil slick images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **2017**, *10*, 5485–5494.

11. Singha, S.; Ressel, R.; Velotto, D.; Lehner, S. A Combination of Traditional and Polarimetric Features for Oil Spill Detection Using TerraSAR-X. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **2016**, *9*, 4979–4990.

12. Topouzelis, K.; Psyllos, A. Oil spill feature selection and classification using decision tree forest on SAR image data. *ISPRS Journal of Photogrammetry and Remote Sensing* **2012**, *68*, 135–143.

13. Taravat, A.; Oppelt, N. Adaptive Weibull Multiplicative Model and Multilayer Perceptron Neural Networks for Dark-Spot Detection from SAR Imagery. *Sensors* **2014**, *14*, 22798–22810.

14. Mera, D.; Bolon-Canedo, V.; Cotos, J.; Alonso-Betanzos, A. On the use of feature selection to improve the detection of sea oil spills in SAR images. *Computers and Geosciences* **2017**, *100*, 166–178.

15. Xu, L.; Li, J.; Brenning, A. A comparative study of different classification techniques for marine oil spill identification using RADARSAT-1 imagery. *Remote Sensing of Environment* **2014**, *141*, 14 – 23.

16. Chehresa, S.; Amirkhani, A.; Rezairad, G.; Mosavi, M. Optimum Features Selection for oil Spill Detection in SAR Image. *Journal of the Indian Society of Remote Sensing* **2016**, *44*, 775–886.

17. Frate, F.; Petrochi, F.; Lichtenegger, J.; Calabresi, G. Neural networks for oil spill detection using ERS-SAR data. *IEEE Transactions on Geoscience and Remote Sensing* **2000**, *38*, 2282–2287.

18. Singha, S.; Bellerby, T.; Trieschmann, O. Satellite oil spill detection using artificial neural networks. *IEEE Journal of selected topixs in applied earth observations and remote sensing* **2013**, *6*, 2355–2363.

19. Zhang, Y.; Li, Y.; Liang, X.; Tsou, J. Comparison of Oil Spill Classifications using Fully and Compact Polarimetric SAR Images. *Applied Sciences* **2016**, *7*, 193:1–22.

20. Topouzelis, K.; Karathanassi, V.; Pavlakis, P.; Rokos, D. Detection and discrimination between oil spills and look-alike phenomena through neural networks. *ISPRS Journal of Photogrammetry and Remote Sensing* **2007**, *62*, 264–270.

21. Topouzelis, K.; Karathanassi, V.; Pavlakis, P.; Rokos, D. Dark formation detection using neural networks. *International Journal of Remote Sensing* **2008**, *29*, 4705–4720.

22. Taravat, A.; Del Frate, F. Development of band ratioing algorithms and neural networks to detection of oil spills using Landsat ETM+ data. *EURASIP Journal on Advances in Signal Processing* **2012**, *107*, 1–8.

23. Chen, G.; Li, Y.; Sun, G.; Zhang, Y. Application of Deep Networks to Oil Spill Detection Using Polarimetric Synthetic Aperture Radar Images. *Applied Sciences* **2017**, *7*.

24. Guo, H.; Wu, D.; An, J. Discrimination of Oil Slicks and Lookalikes in Polarimetric SAR Images Using CNN. *Sensors* **2017**, *17*, 1837:1–20.

25. Oprea, S.O.; Gil, P.; Mira, D.; Alacid, B. Candidate Oil Spill Detection in SLAR Data - A Recurrent Neural Network-based Approach. Proceedings of the 6th International Conference on Pattern Recognition Applications and Methods (ICPRAM), 2017, Vol. 1, pp. 372–377.

26. Ziemke, T. Radar image segmentation using recurrent artificial neural networks. *Pattern Recognition Letters* **1996**, *17*, 319 – 334.

27. Alacid, B.; Gallego, A.J.; Gil, P.; Pertusa, A. Oil Slicks Detection in SLAR Images with Autoencoders. *Proceedings* **2017**, *1*, 820:1–15.

28. Hinton, G.E.; Zemel, R.S. Autoencoders, Minimum Description Length and Helmholtz Free Energy. Advances in Neural Information Processing Systems, 1994, pp. 3–10.

29. Baldi, P. Autoencoders, Unsupervised Learning, and Deep Architectures. JMLR W&CP, 2012, Vol. 27, pp. 37–49.

30. Wang, W.; Huang, Y.; Wang, Y.; Wang, L. Generalized Autoencoder: A Neural Network Framework for Dimensionality Reduction. The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2014, pp. 490–497.

31. Vincent, P.; Larochelle, H.; Lajoie, I.; Bengio, Y.; Manzagol, P.A. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research* **2010**, *11*, 3371–3408.

32. Vincent, P.; Larochelle, H.; Bengio, Y.; Manzagol, P.A. Extracting and Composing Robust Features with Denoising Autoencoders. Proceedings of the 25th International Conference on Machine Learning; ACM: New York, NY, USA, 2008; ICML '08, pp. 1096–1103.

33. Bengio, Y.; Yao, L.; Alain, G.; Vincent, P. Generalized denoising auto-encoders as generative models. Advances in Neural Information Processing Systems 26, 2013, pp. 899–907.

34. Zhao, F.; Liu, Y.; Huo, K.; Zhang, S.; Zhang, Z. Radar HRRP Target Recognition Based on Stacked Autoencoder and Extreme Learning Machine. *Sensors* **2018**, *18*, 173:1–15.

35. Kang, M.; Ji, K.; Leng, X.; Xing, X.; Zou, H. Synthetic Aperture Radar Target Recognition with Feature Fusion Based on a Stacked Autoencoder. *Sensors* **2017**, *17*, 192:1–16.

36. Liang, P.; Shi, W.; Zhang, X. Remote Sensing Image Classification Based on Stacked Denoising Autoencoder. *Remote Sensing* **2018**, *10*, 16:1–12.

37. Gallego, A.; Calvo-Zaragoza, J. Staff-line removal with selective auto-encoders. *Expert Systems with Applications* **2017**, *89*, 138–148.

38. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444.

39. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *JMLR W&CP* **2015**, *37*.

40. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* **2014**, *15*, 1929–1958.

41. Glorot, X.; Bordes, A.; Bengio, Y. Deep Sparse Rectifier Neural Networks. *Journal of Machine Learning Research (JMLR) W&CP* **2011**, *15*, 315–323.

42. Mao, X.; Shen, C.; Yang, Y. Image Restoration Using Very Deep Convolutional Encoder-Decoder Networks with Symmetric Skip Connections. Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain, 2016, pp. 2802–2810.

43. Bergstra, J.; Bengio, Y. Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research* **2012**, *13*, 281–305.

44. Bottou, L. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*; Springer, 2010; pp. 177–186.

45. Zeiler, M.D. ADADELTA: An Adaptive Learning Rate Method. *CoRR* **2012**, *abs/1212.5701*.

46. Alacid, B.; Gil, P. An approach for SLAR images denoising based on removing regions with low visual quality for oil spill detection. SPIE Remote Sensing. International Society for Optics and Photonics, 2016, pp. 1000419–1000419.

47. Kohavi, R. A Study of Cross-validation and Bootstrap for Accuracy Estimation and Model Selection. Proceedings IJCAI; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 1995; Vol. 2, *IJCAI'95*, pp. 1137–1143.

48. Shalabi, L.A.; Shaaban, Z.; Kasasbeh, B. Data Mining: A Preprocessing Engine. *Journal of Computer Science* **2006**, *2*, 735–739.

49. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. *Advances In Neural Information Processing Systems* **2012**, pp. 1–9.

50. Chatfield, K.; Simonyan, K.; Vedaldi, A.; Zisserman, A. Return of the Devil in the Details: Delving Deep into Convolutional Nets. British Machine Vision Conference, 2014, pp. 1–11.

51. Mira, D.; Gil, P.and Alacid, B.; Torres, F. Oil Spill Detection using Segmentation based Approaches. Proceedings of the 6th International Conference on Pattern Recognition Applications and Methods - Volume 1: ICPRAM„ 2017, pp. 442–447.

52. Felzenszwalb, P.; Huttenlocher, D. Efficient Graph-Based Image Segmentation. *International Journal of Computer Vision* **2004**, *59*, 167–181.

53. Deng, Y.; Manjunath, B.S. Unsupervised segmentation of color-texture regions in images and video. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2001**, *23*, 800–810.

54. Isola, P.; Zhu, J.Y.; Zhou, T.; Efros, A.A. Image-to-Image Translation with Conditional Adversarial Networks. *arxiv* **2016**.

55. Liu, Y.; Weisberg, R.H.; Hu, C.; Zheng, L. Tracking the Deepwater Horizon Oil Spill: A Modeling Perspective. *Eos, Transactions American Geophysical Union* **2011**, *92*, 45–46.