Ultra-narrow Band Satellite Visual Monitoring of High-sea Fishing Vessels

Pedro Miguel Garcia Perdigão
Electrical and Computer Engineering
Instituto Superior Técnico
Lisbon, Portugal
pedro.perdigao@ist.utl.pt

***Abstract*— Overfishing, i.e. fishing more fish than can be reproduced, has extremely harmful consequences, affecting the natural balance of the oceans as well as the economic and social well-being of the coastal communities that depend on it for survival. In order to promote the sustainability of marine life, several monitoring techniques have been employed over the years. One of these monitoring techniques is to place a 'blackbox' in the fishing vessels which, via satellite communication, transmits relevant information, such as the boat position and speed, to a Control Center at shore. To increase the effectiveness of this type of monitoring, the implementation of an on-board camera has been subject of study to enable video surveillance on board of fishing vessels. However, the extraordinarily high cost of satellite-based communications is a major barrier, seriously hindering the implementation of such systems.**

In this context, this work has two main objectives: the first concerned the development of an automatic detection system of fishing activities based on visual data. This system was based on the increasingly popular deep-learning networks, notably using the so-called Convolutional Neural Networks to provide video classification. The resulting automatic detection of fishing activities proved to have an excellent performance, thus allowing to alert the occurrence of illegal fishing activities, as well as the transmission of relevant images of those events. The second objective was the efficient compression of the images to be satellite transmitted, notably using a royalty free codec. To increase the final user experience, an image enhancement system to be placed in the Control Center was also developed, to allow performing a lower rate based image compression at the vessels and consequently reducing the transmission costs.

***Keywords*— Overfishing, Vessel Monitoring Systems, Video-surveillance, Image Processing, Image Compression, Deep-Learning, Convolution Neural Networks, Image Enhancement**

I. INTRODUCTION

For centuries, fishing has taken a part on many people's lives that relied on this source of food to survive. If at the beginning our seas and oceans were thought as a limitless source of food, today the reality is much different. In the mid of the last century, international efforts were made to overcome protein-rich food scarcity, which led to a continuously increase of fishing capacity. As commercial fleets aggressively scoured the oceans with unregulated ever growing, sophisticated fishing methods, with nothing but profit in mind, fishing came to a point where it started to be more and more unsustainable.

Faced with the collapse of fish stocks, commercial vessels kept expanding deeper into the ocean, discovering new fish stocks to exploit. By 1996, where fishing peaked at 86.4 million tons [1], few were the fisheries left to be discovered, which led to a decline in catches – not due to countries fishing less, but because fisheries had been exhausted, one after the other. To stop overfishing and its consequences, governments started to enforce fishing-policy rules, aiming to collect the necessary data for managing fishing opportunities, thus being able to ensure that only a certain quantity of fish is caught. The rules must be applied to all fishing vessels (above a certain length) – uncompliant vessels should be heavily sanctioned.

As technology evolves, new ways of enforcing such fishing-policy rules are developed. In fact, a Portuguese company name *Xsealence* was responsible for developing the first Vessel Monitoring System (VMS) back in 1988 [2]. This system, whose operation and architecture will be covered later in this work, is based on a "black box" placed on the fishing vessel that is able to acquire the boat position and other statistics and transmitting them to a Control Center. Currently, *Xsealence* is studying the possibility of adding a video camera to its "black box", to provide the Control Center with the ability to monitor vessels through the acquisition and processing of visual data.

There are, however, complex problems inherent to such a visual-based solution. For instance, while mobile communications are getting increasingly faster and cheaper, in the high-seas the mobile network coverage is virtually nonexistent. As such, the only solution is to use satellite communications. However, the available channel bandwidth is very narrow and expensive which poses an obvious problem to video transmission, as even very low bitrate video would completely saturate the available channel, at prohibitively high costs. Nevertheless, there should be a way to at least transmit a cleverly selected set of images, providing that the user is willing to pay a reasonable extra cost.

II. Still Image Codecs: A Review

This section will briefly review the most relevant still images codec available, since these are the most suited for the context of this work. It will start with JPEG compliant codecs and then move on to video intra-coding solutions (e.g. VP9 Intra) which, currently, are those offering the best rate-distortion (RD) performance.

- **JPEG on Steroids**: one of still image coding solutions proposed to Image Compression Gran Challenge organized in the context of the International Conference on Image Processing 2016 [3]. It uses several techniques to provide better quality for the same rate.
- **Guetzli**: a new open-source perceptually-guided JPEG codec developed by Google and released in 2017. It is stated to be able to produce JPEG images with file sizes 35% smaller than currently available JPEG codecs.
- **JPEG 2000**: JPEG 2000 is an image coding standard developed by the Joint Photographic Experts Group (JPEG) and released in 2000. It was intended to address several limitations of the original JPEG standard.
- **Daala**: Daala is the name of a relatively new video coding technology, which resulted from a collaboration between the Mozzilla Foundation, Xiph.Org Foundation and others.
- **VP9**: VP9 is an open and royalty free video codec that follows its predecessor VP8, which was developed in the WebMproject and further developed by Google. It was designed to address overall online video consumption growth and H.264/AVC and VP8's bandwidth limitations, while attempting to compete with the HEVC video codec, notably for ultra-high resolution videos.
- **Thor**: Thor is a video codec that has been developed by Cisco Systems. Its structure is similar to modern video codecs, although with less features than HEVC.

A comparison between all codecs on both RD performance and computationally complexity is presented in Figure 1.

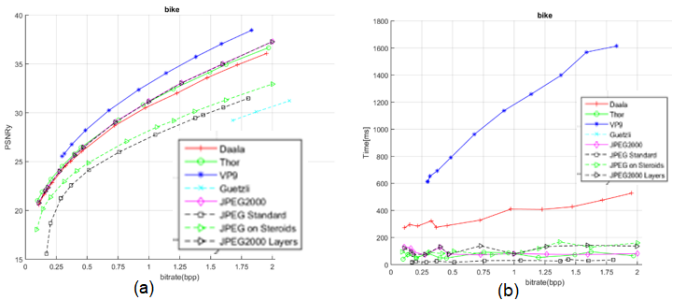

***Figure 1** – Image compression using all described codecs plus a non-optimized JPEG codec, for comparison; (a) RD performance results; (b) Computational complexity results*

An analysis on Figure shows that the best codec is VP9. In second place comes JPEG2000 and Thor, closely followed by Daala. As expected, JPEG compliant codecs provide the worst results. However, JPEG on Steroids, using novel techniques is able to surpass non-optimized JPEG. The fact that its computational complexity is very low and by being compliant with an enormous amount of JPEG decoders, make it a good solution. On the same note, JPEG2000, by providing very good RD performance with very low computational compexiy, makes this codec also a good solution.

III. Fishing Activity Detection and Classification Solution

This Section proposes an automatic solution for the detection and classification of fishing activities based on the processing of real-time video information acquired with a camera well positioned in the vessel.

A. Video Dataset and Ground Truth Definition

Naturally, the design and assessment of a fishing detection and classification solution can only be made if appropriate and representative test material is available. To gather enough video data, a camera was installed in *Noruega*, a fishing vessel used by Instituto Português do Mar e da Atmosfera (IPMA) to perform sea expeditions. Over 100 hours of footage was acquired. Video footage was divided in non-relevant segments and relevant segments, where relevant segments are related with fishing events (i.e. throwing and pulling the net from the sea) and non-relevant segments with all other events (see Figure 2).

***Figure 2** – Two video examples: one without relevant segments/events and another with a single relevant segment/event.*

For relevant, the ground-truth was defined as follows:

- **Throwing the net into the sea**: The start of this task is here associated to actions performed with the net targeting its launching into the sea. The throwing net task is considered finished when the net is no longer visible, as fully under water;
- **Pulling the net from the sea:** The start of this task is here associated to the visibility of the fishing net on the boat deck and thus the pulling out start corresponds to the first frame where the fishing net is minimally visible. The pulling net task is considered finished when the net is totally removed from the boat deck, together with any fish.

***Figure 3** – Two video examples: one without relevant segments/events and another with a single relevant segment/event.*

All other parts of the video, including the time the net is underwater, are defined as non-relevant.

B. Architecture and Walkthrough

The Fishing Activity Detection and Classification engine is responsible for the core functionality of this system, i.e., the detection and classification of fishing events. The architecture of the system is described in Figure 4.

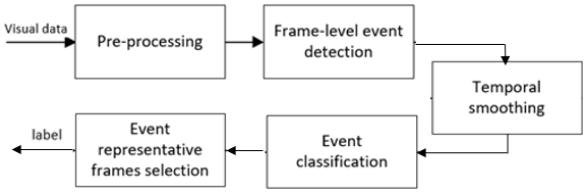

***Figure 4** – Fishing Activity Detection and Classification architecture.*

In this module, the captured visual data is analyzed to assess if a fishing event is taking place or not at some stage. In this section, the architecture and walkthrough of this engine is presented, while the key module, this means the Frame-level event detection module, is presented in detail in the next sub-section.

- **Pre-processing**: Since the next Frame-level event detection module, which is based on deep learning, requires the input data to have certain characteristics, this module has the main task to pre-process the acquired data to convert it to the appropriate data for the next module, notably applying cropping, resizing and mean subtraction;
- **Frame-level event detection**: This module is responsible for individually processing each frame captured by the camera and, on a frame-by-frame basis, computing the probability of a fishing event taking place at that moment. This is achieved using a Deep Learning based approach, more precisely, a Convolutional Neural Network (CNN). This type of solution was chosen as it has been recently proved to provide excellent performance in terms of image detection, classification and recognition related tasks
- **Temporal smoothing**: As the output of the previous module results from a frame-by-frame approach, and thus does not take into account the temporal evolution by using past and future frames, it may happen that these probabilities show rapid and large fluctuations, which is undesirable in terms of event detection and classification. Thus, a temporal filtering operation is performed to smooth the probability-based detection output of the previous module, thus allowing to obtain a more stable and accurate classification, since the fishing events cannot successively change at frame level. This is achieved by using a sliding window with a specific sliding window size, *Sliding_window_size*, measured in seconds. The optimal value for this parameter will be later assessed.
- **Event classification**: The goal of this module is to classify the content in terms of fishing events after assessing whether a relevant event, this means fishing activity, is happening or not, naturally based on the output of the Temporal smoothing module.
  The event classification process where relevant fishing events are detected and classified is based on the so-called Event Detection Ratio (EDR), which is the ratio between the so-called Area Under the Curve (AUC), where the curve is here the filtered frame-level event probabilities within a given time interval (portraited in blue in Figure 9) and the maximum possible area in that same time interval (portraited in grey in Figure 9). Thus, EDR comes:

$$EDR = \frac{AUC}{max\_area} \quad (1)$$

  When, EDR, at some point becomes larger than a pre-defined event threshold, the event_threshold, and thus the event starts to be considered a candidate for a relevant event; when EDR is back to less than event_threshold, the current candidate event is considered finished and finally classified as a relevant event if longer than a pre-defined minimum length (see Figure 5). This minimum length shall be called minimum_relevant_event_duration and has been chosen to be of 1 minute, as all relevant events have a duration of more than 1 minute.

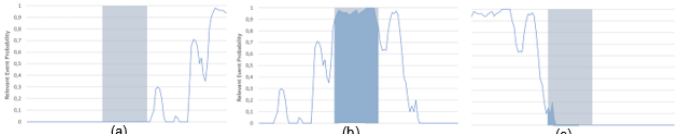

Figure 5 *– Two video examples: one without relevant segments/events and another with a single relevant segment/event.*

- **Event representative frames selection** – After detecting a relevant event, the frame with the highest filtered probability within the last interval corresponding to half of the *event_time_window* parameter is selected and transmitted.

C. Frame-level Fishing Event Detection

This is the key module in the overall architecture with the objective to detect fishing events at frame level. Inspired by the success of Convolutional Neural Networks in image classification, some video classifications approaches have been using CNNs. Although video classification may imply the exploitation of features involving time, it has been shown that this temporal exploitation might not play such an important role and video may be also treated as a sequence of frames. In [4] several approaches to video classification are analyzed to conclude that the best approach is one using information across several frames, which achieves a classification accuracy of 60.9% on top-1 predictions. However, a single-frame approach, which consists in simply forward-passing a frame through a regular CNN architecture, thus not exploiting temporal information, achieved an accuracy of 59.3% on top-1 accuracy predictions. The very small difference between the two performances suggests that motion might not play such a significant role, at least for some classification tasks. After analyzing our fishing detection problem, it was concluded that this may be the case for our problem as the key information is not related to the temporal evolution of the moving elements in the scene but rather of the spatial positions of key elements like the fishing net and the boat doors. Based on this analysis, it was decided to adopt a single-frame approach to initially attribute a fishing event detection probability to each frame. For this, an available CNN classification architecture needs to be chosen as accomplished in the next section.

1) CNN Architecture Selection

While current state-of-the-art CNN solutions achieve exceptionally good results, these are often very computationally expensive. Hence, since the solution is essential to be of low complexity, a comparison comparing several well-kown CNNs is represented in Table 1, showing number of layers involved, top5-error and task running time.

Table 1 *– CNN benchmarkings.*

CNN	Layers	Top-5 error	Running time (ms)
AlexNet [5]	8	18.20%	5.04
Inception-V1 [6]	22	10.07%	12.06
VGG-19 [7]	19	9.00%	55.75
ResNet-152 [8]	152	4.49%	75.45

From Table 1, it is possible to conclude that the CNN architecture offering lower computational complexity is AlexNet [11]. This was expected since its architecture is much

simpler than the studied alternatives as it includes only 8 layers. Although achieving a lower accuracy performance, AlexNet can still do a remarkable job in image classification. In conclusion, AlexNet is the CNN architecture selected for the detection and classification solution to be developed in this work.

2) AlexNet Architecture Overview

Now that a specific CNN architecture has been chosen, it is time to provide some insight about it, starting with the architecture which is presented in Figure 6.

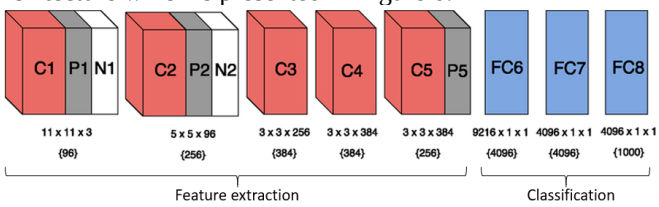

Figure 6 *– AlexNet architecture: C means convolutional layer; P means pooling layer; N means local response normalization layer; FC means fully connected layer..*

As shown in Figure 6, AlexNet includes several types of layers, notably convolutional, pooling, local response normalization and fully connected layers.
As all CNNs, the AlexNet network is divided into two main parts: the feature extraction part and the classification part:

- **Feature extraction part** – Essentially composed by convolutional layers, this part is responsible for extracting the most relevant image features. These layers are usually followed by other layers such as pooling layers and activation layers, which improve the network operation. The feature extraction part works by extracting low-level features in its first layers; as the network progresses, filters are applied to the output of previous layers, therefore progressively extracting higher-level features.
- **Classification part** – This part is essentially composed by fully connected layers and it is responsible for combining the extracted features to model more complex patterns, until reaching the final where a class is attributed to the image.

3) AlexNet Training

This process of training requires many different images for it to converge and generalize properly. However, our available dataset is rather limited, as it was taken from a single boat and thus the features present are always the same. In fact, very few people train an entire CNN from scratch, simply because the availability of datasets large enough to provide proper generalization is scarce [16]. One way to solve this problem is to use transfer learning, where a network trained on a very large dataset is used for a slightly different task by re-training and thus fine-tuning some layers using some additional content. Since the authors [4] achieved the best performance by fine-tuning the layers present in the classification part of the network, this will be also the procedure adopted in this work. This corresponds in practice, in training the classification part of the network, somehow assuming that the extracted features are appropriate even if the classification task changes slightly. However, the classification layers have to be re-trained as the labels are different and thus the extracted features have to be combined in a different way.

For the process of training, the following choices were made: tensorflow was used as deep learning framework; the pre-trained weights were taken from Model Zoo and converted with *Caffe to Tensorflow*; a number of classes to train the network was 2, non-relevant events and relevant events; dropout probability used was of 0.5; batch size was of 128; the loss function used was the cross entropy loss function, the learning rate used was 0.001; the network was trained for 10 epochs.

D. Performance Assessment

This section targets to assess the performance of the fishing events detection and classification solution proposed in the previous sections. To achieve this objective, the process of defining the material used for training and testing will be presented as well as the adopted performance metrics. The performance of the solution will then be assessed.

1) Training and Test Material

Out of the 242 videos from the fishing dataset, 176 do not contain any relevant events, and 66 do. However, only a small portion of each video with relevant events is, in fact, a relevant event, and no video has more than one relevant event.
To train a CNN, each class must have more or less the same images. As such, to build the training and test material, videos with relevant events were randomly divided: approximately 80 % of the frames (without extracting them yet from the videos) were to the training set and the relevant frames extracted; the remaining were to the test set. Then relevant frames were extracted. From the remaining non-relevant segments all frames were extracted and put in the training set. Non-relavant videos were distributed between the training and testing set in a way that the training set had 80% of non-relevant frames. Then, from the non-relevant training set frames, frames were discard until the number of relevant frames and non-relevant frames from the training set was the same. This resulted in 266 778 non-relevant and 266 778 relevant frames for the training procedure.

2) Performance Assessment Metrics and Procedures

To assess the performance of the proposed solution as detailed in Section 4.3, some performance metrics are required. This section defines the metrics that are appropriate to measure the performance of the proposed solution.

a) Receiver Operating Characteristic Curve

A Receiver Operating Characteristic (ROC) curve is a graphical representation of the accuracy of a binary classifier as its discrimination threshold is varied [9]. In this context, a binary classifier is an algorithm with the purpose of, given a stimulus, classifying it as either a “positive event”, i.e. the type of event for which the algorithm was designed to detect, or a “negative event”, i.e. all other uninteresting events. The ROC curve allows to assess the accuracy of a binary classifier (or several) by plotting the True Positive rate versus the False Positive rate for several discrimination thresholds, for some testing set, as depicted in Figure 12 (b). A lower, more lenient discrimination threshold leads to a larger TP rate but at the cost of FP rate increase, and vice-versa. This fundamental tradeoff in these two components will be naturally different from one classifier to another. The aim is to have high TP rate starting as low as possible in terms of FP rate, i.e. a high-performance classifier should have its ROC curve closer to the

TP rate axis getting closer to the maximum TP rate as fast as possible, as seen in Figure 7.

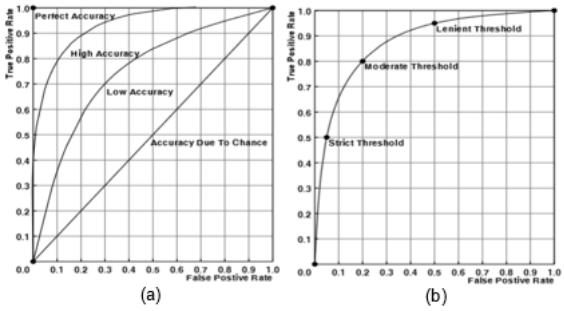

Figure 7 *– ROC curves examples [23]: (a) ROC curves of classifiers with different accuracies; (b) discrimination threshold variation within a ROC curve*

Frame-level Performance Assessment Procedure

In this section, the assessment procedure is performed at the frame level, thus evaluating the output of the architectural module working at this level. More precisely, the performance impact of the temporal filter for several values of the *sliding_window_width* parameter will be studied, therefore choosing the value offering the best performance. To achieve this, the true positives are defined as frames that, for a given discriminative threshold, are classified as part of a relevant event and are indeed part of a relevant event according to the ground-truth and false positives are defined as frames that, for a given discriminative threshold, are classified as part of a relevant event but are, in fact, not part of a relevant event according to the ground-truth.

Event-level Performance Assessment Procedure

The final output of the fishing activity detection and classification framework (in terms of detection and classification, neglecting the representative frame selection) needs also to be evaluated with a suitable procedure. As this output is event based and not directly aligned to the ground-truth data, some delay in the detection of the event can occur or even some misclassification during a relevant (or non-relevant) event. Due to this misalignment and the need to precisely define what is a true/false positive/negative for the event-level performance assessment, it is proposed to divide, both the ground-truth and the output of the classifier, into temporal segments. For this, each test video is divided into equal *segments* of 60 seconds. Then, the ground-truth error boundaries will be rounded to the nearest interval bound, as suggested in Figure.

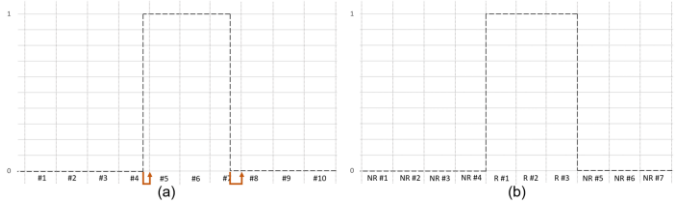

Figure 8 *– Creation of segments and approximation of boundaries for ground-truth. (a) before; (b) after.*

These two steps allow the boundaries of the relevant events in the output of the detection and the ground-truth to be represented with the same temporal precision and thus aligned despite some temporal localization (quantization) error.

True positives can be defined as event segments that, for a given event_threshold, the event detection module classifies as relevant and are indeed relevant according to the ground truth, and false positives can be defined as event segments that, for a given *event_threshold*, the event detection module classifies as relevant but are, in fact, not relevant according to the ground truth.

b) Event Boundaries Localization Error

The previously described metrics are able to describe how well events can be detected. However, it is also important to measure the precision of the boundaries that separate both types of events. Therefore, a procedure to measure the boundaries errors between the detected events and the ground-truth's relevant events is proposed. Two boundary metrics are presented:

possible to measure the following metrics to assess the temporal precision of the detected events:

- **Mean Absolute Boundaries Error (MABE)** – The purpose of the Mean Absolute Boundaries Error metric is to assess how precise are the detected event boundaries in comparison to the ground-truth. This metric is based on the average of the absolute errors between the detected event(s) boundaries and the corresponding ground-truth's relevant event, as shown in (4.5):

$$MABE = \frac{1}{2 \cdot \text{gt\_rel\_events}} \sum_{j=0}^{\text{gt\_rel\_events}} \left(|e_{left,j}| + |e_{right,j}|\right) \qquad (2)$$

Where where gt_rel_events is the total number of ground-truth's relevant events for which boundary errors were successfully computed, and e_(left,j) and e_(right,j) are the errors at the left and right boundary, respectively.

- **Mean Relative Boundaries Error (MRBE)** – Like the MABE metric, the Mean Relative Boundaries Error also evaluates how precise are the detected event boundaries, but in this case in a relative way, thus considering the duration of the relevant event in question. This allows understanding how precise is the proposed solution relatively to the duration of the event, as shown in (4.6):

$$MRBE = \frac{1}{2 \cdot \text{gt\_rel\_events}} \sum_{j=1}^{\text{gt\_rel\_events}} \left(\frac{|e_{left,j}| + |e_{right,j}|}{\text{gt\_event\_duration}_j}\right) \cdot 100 \qquad (3)$$

3) Detection and Classification Results Analysis

In this section, the previously defined performance assessment metrics are used to evaluate the performance of the proposed solution and to find the best values for some relevant parameters. The performance assessment is made at the frame-level and then at the event-level.

Frame-based Performance Assessment

As shown in previously, the proposed solution performs a frame-level classification without exploiting past or future frames, which can lead to a rather inconsistent and highly fluctuating detection output. To address this problem, the output probability of the frame-level classification is used by a temporal smoothing filter that is controlled by the sliding_window_width parameter: the highest its value, the smoother the signal at its output.

To find the optimal value for the sliding_window_width parameter, the smoothed output probability is thresholded and the accuracy is measured at the frame level. Thus, several thresholds were applied, in which frames with a probability higher than a threshold are considered positives, this means a

frame belonging to a relevant event, and frames with a probability lower than a threshold are considered negatives and thus non-relevant. The following thresholds were applied: 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 and 1.

As shown in Figure 9, several ROC curves (this TPR versus FPR metric) were obtained for several sliding_window_width parameter values, notably 0 (meaning that no smoothing is applied), 5, 20, 40 and 60 seconds.

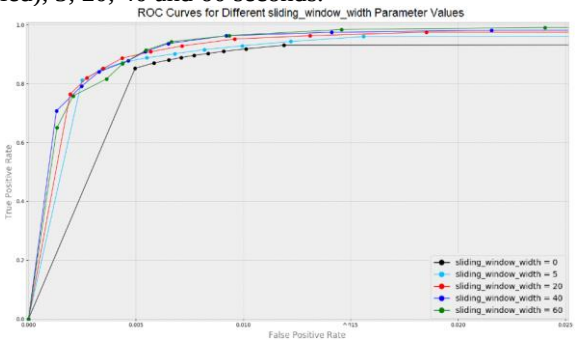

Figure 9 *– ROC curves for different sliding_window_width parameter values*

From Figure 15, it can be concluded that high performance was obtained for the sliding_window_width parameter values of 20, 40 and 60 seconds, with slightly different tradeoffs in terms of TPR/FPR. The value of 20 seconds has been selected as the optimal one as it allows to lower the delay imposed by the temporal smoothing module, and it also allows to obtain (slightly) better TPR/FPR trade-offs for stricter thresholds (0.5 and up).

From the ROC curve, the performance of the frame-level detection module can be evaluated. However, to complement this analysis, the output probability of the temporal smoothing module for some test videos is shown next, for the two types of test videos in this dataset: relevant and non-relevant. A few examples of the behavior on relevant events is displayed in Figure 10, as well as in Figure 11 (a), where a spike of the relevant event probability wrongly occurs. Figure 11 (b) shows why: the maneuvering of the fishing net fools AlexNet into thinking something relevant is occurring. The same happens in Figure 12.

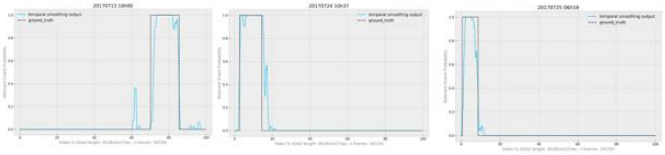

Figure 10 *– Smoothed frame-level relevant event probability for some relevant test videos.*

Figure 11 *– A relevant testing video results: (a) output of the temporal smoothing module; (b) frames extracted from each of the (somewhat) problematic relevant testing videos*

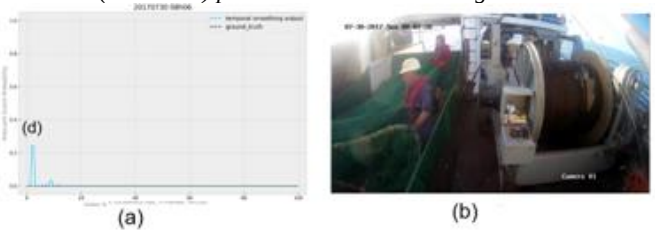

Figure 12 *– A non-relevant testing video results: (a) output of the temporal smoothing module; (b) frames extracted from each of the (somewhat) problematic non-relevant testing videos*

Event-based Performance Assessment

The objective of this section is to study the performance of the proposed solution, notably for different values of the event_threshold and event_time_window parameters, as well as the overall system performance. The event_time_window refers to the window used to compute the output of the temporal smoothing module (frame level relevant event probabilities) to assess whether a relevant event is taking place or not. A larger event_time_window 'looks' at more of the temporal smoothing module output before making a final decision. As for the event_threshold, it is a key parameter for the final detection decision as it determines if the Area Under the Curve (sum of the probabilities in a event_time_window) is enough (or not) for an event to be detected. The event-level performance assessment is performed with two types of metrics. Note also that the computation of the event-level ROC curves is based not on actual non-relevant/relevant events but on the fixed length segments which divide the non-relevant/relevant events, as previously.

Figure 13 presents several ROC curves for different event_time_window values, notably 30, 45, 60, 75 and 90 seconds. For each event_time_window, different event_threshold values were applied, notably 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 and 1. In a ROC curve, the thresholds vary from the highest value at the left, to the lowest value at the right.

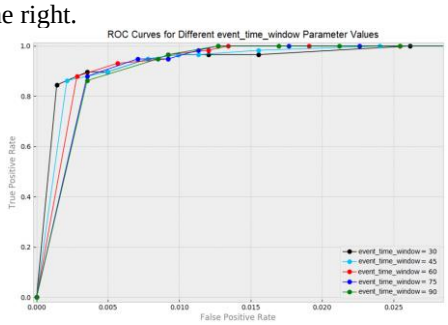

Figure 13 *– ROC curves for different event_time_window values, by varying the event_threshold parameter.*

As shown in Figure 13, the ROC performance for several event_time_window values is rather similar. This can be explained by the good performance of the frame-level event detection and temporal smoothing modules; thus having after a larger or smaller event_time_window does not have a big impact. However, this parameter should be large enough to suppress eventual drops in the relevant event probability and, thus, able to detect a single relevant event reliably, and not two (or more) separate/broken relevant events.

Also, Table 2 shows the results for the two previously presented error boundaries metrics, notably Mean Absolute Boundaries Error and Mean Relative Boundaries Error, for event_time_window parameter values of 30, 45, 60, 75, 90

and event_threshold parameter values of 0.6, 0.7 and 0.8 (for which the TPR/FPR values change more). These metrics measure the event boundary error in absolute and relative ways.

***Table 2** – ROC curves for different event_time_window values, by varying the event_threshold parameter.*

Event_time_window Parameter (seconds)	*Event_threshold* Parameter	MABE (seconds)	MRBE (%)
30	**0.6**	25.6	14.7%
	0.7	23.3	13.3%
	0.8	21.2	11.9%
45	**0.6**	29.3	16.3%
	0.7	24.8	14.1%
	0.8	20.7	11.4%
60	**0.6**	30.4	17.1%
	0.7	27.4	15.6%
	0.8	21.0	11.5%
75	**0.6**	35.0	19.4%
	0.7	29.2	16.5%
	0.8	22.4	12.3%
90	**0.6**	40.1	21.9%
	0.7	31.4	17.6%
	0.8	23.9	13.1%

From the ROC results in Figure 13, the best performance achieved for medium to high FPRs is obtained for a event_time_window value of 60. A trend can be observed in Figure 13 as when event_time_window increases better performance is obtained for medium to high FPRs but the performance for low FPRs decreases. From the results in Table 2, it is possible to conclude that the errors increase (for the same event_threshold) when the event_time_window increases, for most the cases. This was expected since the larger decision windows contribute to a larger imprecision in the definition of the event boundaries. In this case, the event_time_window parameter value of 60 seconds represents a compromise as it does not have a large boundary error (comparing to event_time_window of 45s only increases 5s) and minimizes the number of false alarms comparing to other event_time_window ROC curves.

The event_threshold parameter should be selected according to the application requirements, since it represents a tradeoff between detecting more relevant events at the cost of more false positives and vice-versa.

4) Coding Results and Analysis

As previously seen, the fishing vessel transmits some selected images of fishing events through a satellite link. This type of channel has extremely low bandwidth as well as large delays and high cost. Thus, it is crucial to compress the image before transmission to the Control Center as much as possible. As such, two codecs were selected: JPEG on Steroids described in [10] available in [11], and JPEG 2000 using the OpenJpeg implementation, due to their computational complexity and compression performance. For comparison, a non-optimized JPEG was also used. Table represents how the the PSNR quality metric varies when increasing a quality parameter (quality factor for JPEG and compression ratio for JPEG 2000). A deblocking filter was also used for the JPEG compliant codecs, thus increasing their quality. On the other side Table shows a cost comparison, presenting the cost of transmitting one frame for when compressed by each codec, for a given objective quality.

***Table 3** – Cost comparison between the three codecs.*

		PSNR (dB)	**Size** (bytes)	**Cost $**
Q1	**Non-optimized JPEG**	27.27	2641	2.81
	JPEG on Steroids	27.39	2337	2.49
	JPEG 2000	27.34	1482	1.58
Q2	**Non-optimized JPEG**	29.02	3533	3.76
	JPEG on Steroids	28.98	3433	3.65
	JPEG 2000	29.03	2257	2.40
Q3	**Non-optimized JPEG**	30.57	4822	5.13
	JPEG on Steroids*	30.66	4970	5.29
	JPEG 2000	30.62	3134	3.34

The deblocking filter used in the JPEG compliant codecs help them to achieve a better compression performance. However, this only happens to some extent. At some point it actually starts to degrade the image.

As expected, JPEG 2000 is the one the offers best compression performance, and thus the lowest transmission cost. However, it lacks support in many applications, which might cause compatiubility problems. On the other side, JPEG on Steroids is able to achieve a good performance while being compatible with an enormous number of JPEG decoders. For this reason, both codecs were implemented in the proposed solution, to be used as needed.

IV. Quality Enhancement for Compression Degraded Images

To maintain a low transmission cost from the fishing vessel using satellite channels, the decoded quality is rather low for the adopted coding solutions and thus, this chapter aims at proposing a powerful post-processing stage to be performed at the Control Center where the decoded image is enhanced and significant amount of computational resources are available.

Image enhancement is a classical problem in low-level vision and the associated technologies aim to improve/restore/enhance the overall quality of a degraded/corrupted image. This topic has been widely studied and discussed in the literature and a wide range of solutions have been proposed, notably related to operations such as denoising, inpainting, deblocking, and super-resolution, among others. An example of this is provided in Figure 14.

***Figure 14** – Image degradation process followed by a restoration/enhancement step.*

Such processing operations have been traditionally based on analytical methods, which heavily rely on previous knowledge of the problem (assumptions). With the emergence of deep learning tools, several attempts to address such imaging problems have been made, which have provided very encouraging results, surpassing all state-of-the-art's.

When it comes to lower the image storage/transmission size, two approaches are interesting for this case: i) down-sampling, with the uncompressed image corresponding to a (reduced) number of pixels, and ii) image coding with a lower quality, i.e. employing a larger quantization step size. These processes result in two major types of artifacts: artifacts due to the upscaling interpolation method (considering that the image is up-sampled back to the original resolution) and compression

artifacts due to the lossy compression method applied. The problem of enhancing the image by removing these artifacts has been addressed by the research community using deep learning models, notably using super-resolution and (lossy) compression artifacts removal approaches.

Enhancing by Super-resolution

Generic single image super-resolution (SISR) algorithms aim to generate high-quality high-resolution (HR) images from a single low-resolution (LR) input image. Typically, SISR CNNs take as input an already up-sampled image obtained by a standard interpolation method (usually bicubic) to the target spatial resolution, rather than having the CNN up-sampling the image itself.

Several super-resolution solutions have been introduced. Lately, such solutions have achieved much better performance by a large margin by using residual learning, which allowed accurate training of very deep networks. In [12], a network architecture very similar to the well-known VGG-net with 20 layers was proposed. Similarly, in [13], a 16-layer network was proposed to restore several low-level images artifacts, SISR inclusive. More recently, the performance has increased by using very deep networks with residual blocks as used in the ResNet image classification CNN.

Enhancing by Compression Artifacts Removal

Lossy image compression solutions, e.g. JPEG, WebP, HEVC codecs, are able to achieve high image compression ratios by discarding less perceptually relevant information for the human vision system. However, very high compression ratios lead to complex and highly visible compression artifacts, notably blocking artifacts, ringing effects and blurring [14].

In recent years, some image enhancement solutions based on CNNs have been proposed to remove such compression artifacts. Due to the huge popularity of the JPEG codec, such solutions focus mainly on JPEG compression artifacts. A first successful attempt was achieved in [14] by essentially reusing the CNN architecture used for SISR in [7], with some minor changes. Later, a deeper CNN resorting to residual learning and a multi-scale loss function, i.e. with several loss functions were computed at several points of the network, was introduced. Similar results were obtained with the network proposed in [13]. Interestingly, the latter was trained both for SISR as well as for image denoising, thus providing very good image enhancement generalization capabilities.

A. Adopted Image Enhancement CNN

From the several networks described previously, only the network presented in [13], the DnCNN network, has been evaluated (by its authors) and the results show that works well for both super-resolution and compression artifacts removal (notably JPEGs artifacts) even when trained for different types of artifact removal/denoising in the same model. The reasons behind the performance of these network are threefold:

- **Deep architecture**: This CNN includes 16 convolutional layers;
- **Residual Learning**: This network employs a single residual unit to predict the residual image, meaning that upon receiving a distorted image at its input, the CNN will implicitly remove the clean image throughout its hidden layers, thus producing the distortion at its output which will be then subtracted from the image;
- **Batch normalization**: Allows a faster training, which is especially important for deep networks.

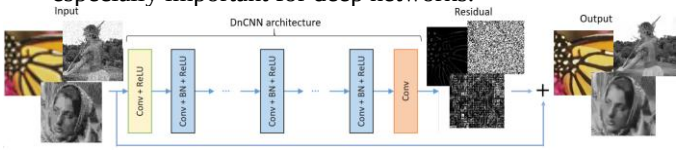

Figure 15 *– DnCNN architecture [9].*

1) Double-Stage DnCNN-based Image Enhancement Configuration

Image storage/transmission size reduction can be obtained by down-sampling, by lossy compression, or both. Using such methods can yield very high compression factors, and thus low rates, regarding the original image but typically at the cost of a poor-quality image. However, using the DnCNN presented in previously some of the lost details can be restored and the overall subjective quality of the image can be improved. Since the compression artifacts that the DnCNN was trained to remove were JPEG related ones, the proposed image enhancement strategy assumes encoding performed with the JPEG on Steroids.

The proposed enhancement configuration aims to enhance a low-quality image which was produced after taking an image with high-resolution, performing spatial down-sampling with a certain pre-selected scaling factor, and then using the JPEG on Steroids encoder to finally compress the image. In the context of this MSc Thesis, this process would take place on the vessel, before transmission. Upon receiving the compressed image, the Control Center would proceed to enhance the image by first decoding the image, and after removing first the compression artifacts (in practice performing deblocking) and next restore the image to its original resolution using a super-resolution strategy (upscaling and DnCNN enhancement). This process is depicted in Figure 16.

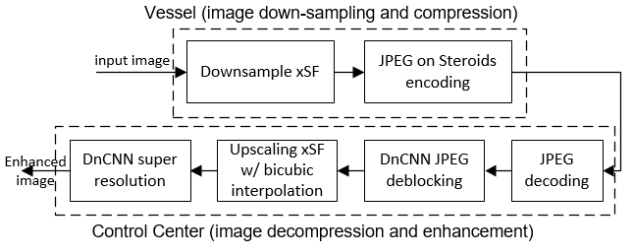

Figure 16 *– Double stage image enhancement configuration. SF means scaling factor.*

The proposed double-stage enhancement configuration is implemented using a single DnCNN model, i.e., only one network trained for both super-resolution and JPEG deblocking is used. The double-stage enhancement strategy presented in **Error! Reference source not found.** targets to remove one artifact at a time. First, the image is down-sampled and coded using the JPEG on Steroids encoder. After decoding with the JPEG on Steroids decoder, the image is passed through de DnCNN for deblocking. Since only JPEG related artifacts are present, the DnCNN removes them as best as possible. Then, after upscaling the image to its original spatial resolution with a bicubic interpolation, a new set of artifacts related to the upscaling of the image appears which are addressed by processing the deblocked and up-sampled image through the DnCNN again to remove these artifacts from the image; since the network was also trained for super-

resolution, it should be able to also (as much as possible) remove these artifacts.

There is, however, an issue that needs careful consideration depending on the application and targets at hand: depending on the desired compression ratio, the optimal down-sampling (and consequent up-sampling) scaling factor differs. For instance, if the goal is to obtain better quality (and thus less compressed) images, then no down-sampling should be applied before coding, as doing so would result in a worse decoded quality image; however, as the need for higher compression increases, down-sampling before coding starts yielding better final quality images. This means that, for a given range of compression ratio/bitrates, there is an optimal scaling factor. The optimal down-sampling scaling factor value for the context of this Thesis will be studied in the following section.

B. Performance Assessment

To asses the performance of the solution, 10 frames extracted from fishing dataset were extracted. These images were resized and cropped to a resolution of 640x360.

1) Single-Stage Image Enhancement Configuration Performance Assessment

To first assess how well the super-resolution and deblocking provided by the DnCNN fair, a comparison between images before and after deblocking are provided in Table 4. As for super-resolution, Table 5 compares bicubic interpolation, Lanczos and DnCNN super-resolution.

Table 4 *– Average values of PSNR, for each codec with and without pos-processing and QP*

	PSNR (dB)			SSIM		
Quality factor (Q)	**No enhc.**	**DnCNN enhc.**	**Gain**	**No enhc.**	**DnCNN enhc.**	**Gain**
Q = 6	26.11	26.92	0.80	0.68	0.73	0.05
Q = 10	27.63	28.54	0.91	0.75	0.79	0.04
Q = 15	28.85	29.92	1.07	0.80	0.83	0.03
Q = 20	29.74	30.92	1.18	0.83	0.86	0.03
Q = 25	30.43	31.64	1.21	0.85	0.87	0.02
Q = 30	31.03	32.28	1.25	0.87	0.89	0.02

Table 5 *– Average values of PSNR, for each codec with and without pos-processing and QP*

	PSNR			SSIM		
Scaling Factor	**Bicubic**	**Lanczos 3**	**DnCNN enhc.**	**Bicubic**	**Lanczos 3**	**DnCNN enhc.**
2x	29.24	29.50	32.13	0.87	0.88	0.93
3x	27.24	27.37	28.93	0.80	0.81	0.86
4x	26.28	26.39	27.51	0.75	0.75	0.81

As expected using the DnCNN for both deblocking and super-resolution provides substantial improvements.

2) Double-Stage Image Enhancement Configuration Performance Assessment

After the individual analysis of the two enhancements approaches, notably deblocking and super-resolution, it is time to present the performance assessment of the previously proposed double-stage image enhancement configuration.

a) Finding the Optimal Down-Sampling Scaling Factor Value

The optimal value for the down-sampling (and consequent up-sampling) scaling factor, here-after referred only as *scaling factor*, depends on the desired bitrate/compression ratio. Thus, to find it, the RD performance with the PSNR and SSIM distortion metrics is presented in Figure 17, for scaling factors of 1 (meaning the image is not resized), 2, 3 and 4. For ease of reading, a Table 6 contains the intervals with the optimal scaling factors for each interval.

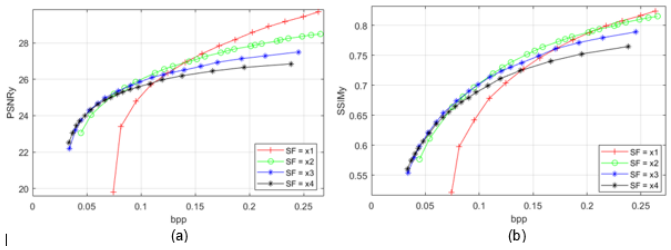

Figure 17 *– Double stage image enhancement configuration. SF means scaling factor.*

Table 6 *– Optimal scaling factors for each bitrate interval by presenting the upper and lower bounds of the intervals where a given scaling factor is optimal.*

		Scaling Factor			
		1x	**2x**	**3x**	**4x**
PSNR	**Upper bound (bpp/kB)**	∞	0.141/ 4.06	0.079/ 2.28	0.048/1. 38
	Lower bound (bpp/kB)	0.141/4.0 6	0.079/ 2.28	0.048/ 1.38	˜∞
SSIM	**Upper bound (bpp/kB)**	∞	0.205/ 5.90	0.095/ 2.74	0.044/ 1.27
	Lower bound (bpp/kB)	0.205/5.9 0	0.095/ 2.74	0.044/ 1.27	˜∞

Table shows discrepancies between the optimal intervals for the PSNR and SSIM metrics for a given scaling factor. For example, PSNR recommends changing the scaling factor from 3 to 2 at a bpp/kB of 0.072/2.28, whereas SSIM recommends doing this at 0.095/2.74. However, as SSIM is a more perceptual-based metric, the optimal scale factors should be chosen according to its defined intervals.

b) Assessing the Contributions of each Image Enhancement Stage

Above, the optimal scaling factor for different bitrates was found for the double-stage image enhancement configuration, which is constituted by two enhancement stages: deblocking and super-resolution. However, it is not yet known the individual contribution of each of these image enhancement stages to the final quality. This will be addressed next.

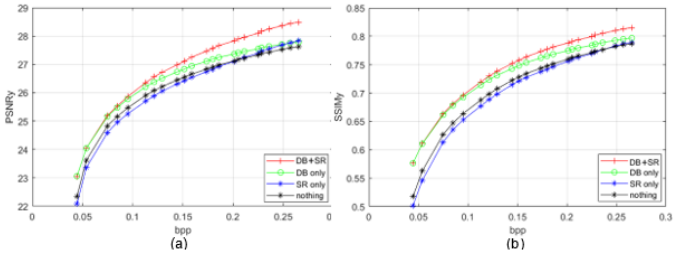

Figure 18 *– RD performance for four different enhancement approaches for a scaling factor of 2 and varying quality factor. The left-most point was obtained using a quality factor of 3; the right-most point was obtained using a quality factor of 61.*

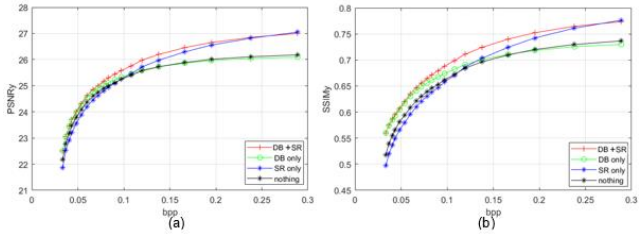

Figure 19 *– performance for four different enhancement approaches for a scaling factor of 4 and varying quality factor. The left-most*

point was obtained using a quality factor of 5; the right-most point was obtained using a quality factor of 96.

The analysis of the results allows to conclude that, for lower bitrates, the double-stage image configuration is essentially the same as when using deblocking only. This was expected since the lack of detail due to very high compression is very intense. Since some pre-existing detail is required to efficiently perform super-resolution, very low bitrate decoded images do not benefit much from super-resolution techniques. As the bitrate increases, so does the detail, thus providing the DnCNN with more information to work with, therefore increasing the super-resolution stage contribution in terms of quality improvements. At some point, for higher quality factors, deblocking actually ends up harming the image quality.

When performing super-resolution only for lower rates the resulting quality is actually worse; since the decoded image is not passed through the DnCNN right after decoding, when the image is upscaled, so are the compression artifacts. When passing the resulting image through the DnCNN, the network is unable to 'understand' the upscaled compression artifacts (they may not have been part of the training) which causes to actually have a worse image quality. As the quality factor values used in the encoding process increase, so does the contribution of this stage.

V. Conclusion

The purpose of this work was to overcome the problem of not knowing when an (illegal) fishing activity is occurring. To solve this, it was developed an automatic solution that, by visually monitoring the vessel activities, is able to automatically detect when fishing activities occur. After detection, the Control Center is also informed of a relevant event followed by receiving visual data that could eventually confirm or deny if any illegal activity is being performed.

To achieve this, an available deep learning based solution using a Convolutional Neural Network developed for image classification was used. The performance assessment of the designed fishing detection solution has shown that it provides very good results in detecting relevant events, i.e., fishing activities visible to the camera.

Another objective was to compress and transmit a fishing event representative frame to the Control Center with as low as possible compressed size (for an acceptable quality) to lower the transmission costs. To achieve this, a study on the most relevant available royalty free codecs and their performance was presented. Also, a solution suitable for enhancing heavy compressed images was developed. This solution was able to enhance compressed images by a significant amount, providing the user with a much better perceptual-quality.

However, there is still room for improvement. The fishing dataset on which the fishing detection tool was trained only has images of the same boat, limiting generalization of the tool. Gathering more footage from different boats would certainly be helpful. Also, the fishing detection tool could use other more powerful (and more computationally complex) algorithms, further improving the fishing detection performance. To finalize, several new image enhancement CNN architectures have been developed. Using more recent and better performing CNNs would certainly image enhancement results.

VI. References

[1] "The State of World Fisheries and Aquaculture," 2016. [Online]. Available: http://www.fao.org/3/a-i5555e.pdf. [Accessed April 2017].

[2] "Fishing Monitoring Systems: Past, Present and Future," [Online]. Available: http://www.imcsnet.org/imcs/docs/fishing_vessel_monitoring_systems_past_present_future.pdf. [Accessed March 2017].

[3] "ICIP 2016," [Online]. Available: http://2016.ieeeicip.org/. [Accessed May 2017].

[4] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and F. F. Li, "Large-scale video classification with convolutional neural networks," in *Computer Vision and Pattern Recognition (CVPR)*, Columbus, Ohio, 2014.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Neural Information Processing Systems (NIPS)*, Lake Tahoe, Nevada, 2012.

[6] C. Szegedy et al., "Going deeper with convolutions," 2014. [Online]. Available: arXiv:1409.4842.

[7] A. Zisserman, K. Simonyan, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *Information and Software Technology*, vol. 51, no. 4, pp. 769-784, 2015.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *Multimedia Tools and Applications*, vol. 77, no. 9, pp. 10437-10453, 2015.

[9] "Wikipedia on Receiver Operating Characteristic," [Online]. Available: https://en.wikipedia.org/wiki/Receiver_operating_characteristic. [Accessed 7 December 2017].

[10] T. Richter, "JPEG on STEROIDS: Common optimization techniques for JPEG image compression," *Proceedings - International Conference on Image Processing, ICIP*, vol. August, pp. 61-65, 2016.

[11] "JPEG on Steroids libjeg release," [Online]. Available: https://github.com/thorfdbg/libjpeg. [Accessed 3 May 2017].

[12] J. Kim, J. K. Lee, and K. M. Lee, "Accurate Image Super-Resolution Using Very Deep Convolutional Networks," in *Conference on Computer Vision and Pattern Recognition*, Nevada, EUA, 2016.

[13] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising," *IEEE Transactions on Image Processing*, Vols. 26, no. 7, p. 3142–3155, 2017.

[14] C. Dong, Y. Deng, C. Loy, and X. Tang, "Compression artifacts reduction by a deep convolutional network," *Proceedings of the IEEE International Conference on Computer Vision*, p. 576–584, 2015.