Learning Temporal Features for Detection on Maritime Airborne Video Sequences using Convolutional LSTM

Gonçalo Cruz, Alexandre Bernardino, *Member, IEEE,*

***Abstract*—In this paper, we study the effectiveness of learning temporal features to improve detection performance in videos captured by small aircraft. To implement this learning process, we use a convolutional Long Short-Term Memory (LSTM) associated with a pre-trained Convolutional Neural Network (CNN). To improve the training process, we incorporate domain-specific knowledge about the expected size and number of boats. We carry out three tests. The first searches the best sequence length and sub-sampling rate for training and the second compares the proposed method with a traditional CNN, a traditional LSTM and a Gated Recurrent Unit (GRU). The final test, evaluates our method with already published detectors, in two datasets. Results show that in favorable conditions, our method's performance is comparable to other detectors but, on more challenging environments, it stands out from other techniques.**

***Index Terms*—Object Detection, Remote Monitoring, Recurrent Neural Networks**

I. Introduction

With 70 percent of our planet covered with water, 90 percent of global trade done by sea [1] and 40 per cent of the world population living near the coast [2], there are strong ecological, economic and social motivations to try to guarantee the safety of people, ecosystems and goods on these regions. Nonetheless, there have been many mishaps over the last decade. Piracy has affected the routes near Nigeria, Somalia and Southeast Asia [3], many migrants have lost their lives on the Mediterranean Sea [4] and ecological disasters affect marine environments [5]. Despite its importance, maritime monitoring remains a challenging task.

The usual approach to maritime surveillance implies the use of satellites, vessels and aircraft (either helicopters or airplanes). Satellites are costly, do not have the flexibility to monitor an arbitrary area at a given time and are not adequate to capture an object of interest in detail. Manned vessels and aircraft usually have radars [6] that allow long-range detection, although some types of ships (especially relevant in smuggling and in search and rescue scenarios) are not easily detected by radar as they are made of wood or plastic [7]. Furthermore, radar cannot be used on-board small size vehicles like Unmanned Aerial Vehicles (UAVs) due to power, weight and space requirements.

G. Cruz is with the Portuguese Air Force Research Center, 2715-021 Sintra, Portugal e-mail: gccruz@academiafa.edu.pt.

A. Bernardino is with the Institute for Systems and Robotics, Department of Electrical and Computer Engineering, Instituto Superior Técnico, 1049-001 Lisboa, Portugal

Fig. 1. Example of a demanding situation, with the boat crossing an area with severe glare. Nevertheless, our method is able to detect the boat in all frames. The top left image also shows a detection of a small size life raft, which is not detected in the other three images due to glare.

The automatic detection based on passive electro-optical sensors, like Sentient's detection system aboard ScanEagle UAV, answers some mentioned difficulties. In particular, it does not depend on the radio wave reflective properties of the objects to detect, and already enabled authorities to detect and intercept smuggling vessels with small radar signature [8] [9]. Detection is one central task in computer vision and has achieved a significant success in general-purpose datasets like COCO [10], where typically the object to detect is predominant on the image. It has also achieved some success in more constrained scenarios, like cameras mounted onshore [11] or on ships [12].

The information extraction from images captured by aerial platforms is more challenging since it is affected by factors like scale, perspective and illumination variations. These images can even change dramatically, depending on the type of UAV that is used. If a small quad-rotor is used [13], the object of interest will appear closer than if we use a fixed-wing aircraft [14]. In this work, we use images captured by a small size fixed wing UAV. On one hand, these aircraft survey an area larger than what is typical for quadcopters but offer a smaller spatial resolution. On the other hand, these images are captured by platforms and sensors much cheaper than satellites or high flying aircraft [15] but are not orthorectified and corrected.

As seen on Fig. 1, the images that we use are affected by phenomena like glare that makes detection difficult for

common detectors. In our previous works, we have already considered this problem. In the first approach, we have designed features useful for this case [16]. The second method used CNNs to learn visual features from similar images [17] and on the third method, we have associated a CNN with a tracker to verify consistency between consecutive frames [18]. Despite achieving interesting results on SEAGULL dataset [14], there are some conditions where these methods fail. Consequently, in this work we introduce a method to robustly detect objects in airborne maritime surveillance images, which are affected by glare, wakes and waves' crests.

A. Contributions

The main contributions of our work can be summarized as follows:

- Characterize the applicability of convolutional LSTM to learn visual and temporal features relevant for detection of maritime objects in airborne video sequences.
- Incorporate Domain Specific Knowledge about maritime objects' size and the number of visible objects at a given time to improve training and the detection performance. In particular, we penalize predictions with large areas labeled as containing a boat.
- Effect analysis of the time scale considered by the neural network on the detection robustness.
- Compare the proposed methodology, in multiple maritime monitoring scenarios, with a mainstream detection network, YOLO [19], and also with one of our previously published methods, that uses visual features and a Multiple Hypothesis Tracker (MHT) to associate detections [18].

B. Outline of the paper

The paper is organized as follows. The next section reviews the literature about object detection and the Section III presents the network's architecture and provides detail about the Conv-LSTM layer and the loss function. In Section IV, we describe the dataset that was used to train and test our method, present the evaluation metrics and the results. In the last section, we conclude this work.

II. Related work

Some of the first applications of vehicle detection from aerial images assumed moving vehicles over land, with some authors using techniques like background subtraction [20]. Background subtraction is effective when there are static objects with features that allow the registration and alignment of consecutive images. In scenarios over the ocean, usually the most distinctive visible features (other than the objects of interest) are glare, waves and wakes, all of which are dynamic phenomena, that change rapidly and hinder image alignment. Other authors assume that an objects' position is limited to areas like roads [21], which is not applicable in maritime surveillance scenarios as vessels can move virtually anywhere.

Another traditional approach to vehicle detection from aerial images was to extract relevant features, from image regions with potential targets, and then classify them with some machine learning technique. The features could be either general purpose like Haar features [22] and features based on Discrete Cosine Transform [23] or specialized ones like color and textures [24] [25]. These approaches are useful for applications with well-defined conditions, however, airborne images captured by small aircraft in maritime scenarios have objects with a large range of sizes, orientations and shapes. To face this challenge, some works have used saliency methods, *i.e.* algorithms that try to emulate the human visual attention mechanism [26] [27]. The mentioned approaches highlight areas that are distinct from the background but that may not correspond to the object of interest. These undesired detections are usually suppressed by using either a heuristic, like checking if a given detection persists on a given number of frames [16] or a more formal framework like the usage of a Hidden Markov Model [28].

Following the advances in computer vision and pattern recognition, maritime detection has also adopted deep learning. Maire *et al.* [29] have proposed the application of CNNs in a sliding window fashion, for the detection of marine mammals. Their relatively shallow network architecture led to limited results. Bousetouane and Morris [30] have also used CNNs for the detection in a maritime scenario. Their pipeline includes several weak detectors to compute candidate regions, extracts features learned by a neural network and then classifies them with a support vector machine. The main downside of this approach is that the first set of weak detectors relies on hand engineered features. While this approach performs well on their case, in our scenario the targets have significant appearance variability which makes the features' manual selection intractable.

There are other approaches, like detection on wide area imagery, which rely on networks like Fast R-CNN [31]. Wide area imagery is especially suited for this kind of networks because it is usually orthorectified and the ground distance corresponding to a pixel is well characterized. This allows to more easily define anchors, which are very relevant for this family of methods. In addition to adaptations of canonical networks there have been inovative approaches for the case of satellite images. For instance, Wang *et al.* have designed a network for change detection, exploring with different weights spectral and subpixel information [32]. Like the previous example, this approach also needs image pre-processing steps. In the present work, the aircraft's movement and the perspective variations hinder the applicability of these preprocessing steps.

There have been some advances using the exploitation of information contained in videos sequences. By consecutively observing a given object in several frames, phenomena like glare and waves are discarded because their persistence is limited in time. Consequently, the correct object can be detected even if, in some frames, it gets occluded or its appearance changes dramatically. Historically, approaches like Markov Chain Monte Carlo data association [33] and MHT [34], have been used to successfully associate detections in consecutive time instants, achieving highly robust detection results. In [18], we have also used MHT to improve the results obtained with a CNN. The downside of this approach is that the movement

model is tuned by the designer and not learned from data. Additionally, the visual features and temporal features are handled separately, which means that if there is a given object which is persistent on the image but the visual cues are not recognized by the neural network, it is not detected.

More recently, some approaches have explored data-driven learning methods, for sequential data, like recurrent neural networks, in particular, the LSTM layer [35]. The core of the LSTM layer is a memory cell that encodes knowledge about the features seen up to a given moment. This cell is able to keep or discard this knowledge due to three gates (input, output and forget) that control the amount of information that enters, exits and is kept on the layer. One interesting LSTM usage was suggested by Wang *et al.* [36], in which the LSTM, associated with a CNN, implements an attention mechanism for scene classification. The recurrent module obtains high level features from the CNN, sequentially generates attention masks and incrementaly classifies the images. Despite being a different task, this indicates that recurrent processing might improve the performance of the considered task.

An interesting approach for image-like data, presented in [37] is the use of LSTM with convolutional structure. Convolutional LSTM (ConvLSTM), as named by its authors, removes the spatial redundancy present in the application of traditional LSTMs to image-like data, much in the same way convolutional neural networks remove spatial redundancy present in fully connected neural networks. Nonetheless, ConvLSTM retains the internal structure (with memory cells, and forget, input and output gates) that allows the common LSTM to keep memory during long periods and using it when relevant. The main difference is that the gates in LSTM are applied multiplicatively and in ConvLSTM are applied in a convolutional fashion.

In our work, our objective is to use ConvLSTM to learn temporal and visual features that improve the detection of boats in video sequences captured by a RGB camera installed on a small size aircraft, in maritime environment. The videos are part of SEAGULL dataset [14] and illustrate realistic maritime monitoring missions. The observed boats have variable sizes and shapes, ranging from small life rafts to high-speed boats to cargo ships. Additionally, the variation in observation perspective also changes the appearance of these boats as shown in Fig. 1. Due to the aircraft's characteristics and its flight pattern, the video contains significant apparent movement. The sequences include large amplitude and low-frequency movements as well as small amplitude and high-frequency components. Additionally, the movement is caused by both linear motion and rotations. Furthermore, the sequences were captured during sunny days, over the Atlantic Ocean. This means that, as exhibited in Fig. 1, the detector has to deal with glare and waves' white caps.

III. Convolutional LSTM Network

A. Problem Description

The present section formally describes the problem considered in this work. We have used videos from SEAGULL dataset (which are detailed in Subsection IV-A) to both train and test our method as they are representative of maritime monitoring missions. In either stage, we extract short video sequences from the full-length videos and we use them as our samples. Since the observed scene does not change significantly between two frames, we pick one image every r^{th} frame, as displayed in Fig. 2. In the rest of the work, we will designate this separation of r frames as time stride. Each of these samples X_t is a 4D tensor, composed of K images, *i.e.* $X_t = \left[x_{t-(K\times r)}, \ldots, x_t\right]$, where x_t is the image captured at instant t. In Fig. 2 (a), we show an example where the number of processed frames is $K = 3$ and the separation between them (time stride) is $r = 3$. In this case, only images x_t, x_{t-3} and x_{t-6}, represented in a darker tone, are considered and the lighter images are discarded.

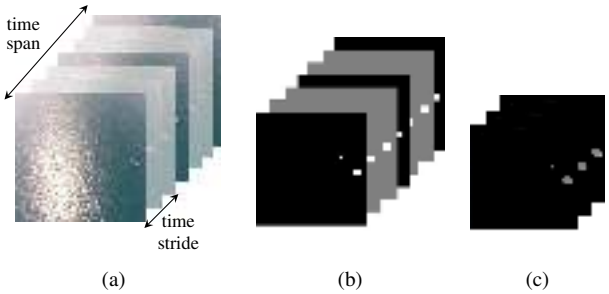

Fig. 2. Example of (a) input X_t, (b) ground truth Y_t and (c) prediction $\hat{Y}_t$ in case the sequence has a time span corresponding to 7 frames but only 3 frames are processed. Images are represented in a lighter tone to indicate that they are discarded.

All videos were manually labeled, marking the location (bounding box) of all objects of interest, and therefore each sequence X_t has a corresponding ground truth Y_t. Analogously to the video sequences, each ground truth Y_t incorporates K ground truth maps, separated r frames between them, $Y_t = \left[y_{t-(K\times r)}, \ldots, y_t\right]$. As shown in Fig. 2(b), a ground truth map y_t consists of a binary image where pixels with *ones* indicate locations with objects of interest. Similarly to videos sequences, in the presented example, we only consider the ground truths y_t, y_{t-3} and y_{t-6}. All images x and labels y were resized to a resolution of 720×720 and 300×300, respectively. The neural network's goal is to obtain an estimate $\hat{Y}_t$ similar to the ground truth, as displayed on Fig.2(c).

B. Convolutional LSTM

The neural network that is used on the present work has different types of layers but we will focus on the impact of one particular type for the detection task. This type of layer is the Convolutional Long Short-Term Memory (ConvLSTM). We have followed an approach similar to Wingjian *et al.* [37], using a modified version of traditional LSTM, where the input-to-state and state-to-state multiplications are replaced by convolutions. Similarly to traditional LSTM, its convolutional alternative contains mechanisms that control the amount of information that is received and outputted by the network. These mechanisms are the *input gate* i_t and *output gate* o_t and its equations are

$$i_t = \sigma(W_{ZI} * z_t + W_{HI} * \hat{y}_{t-1} + W_{CI} \circ c_{t-1} + b_I) \quad (1)$$

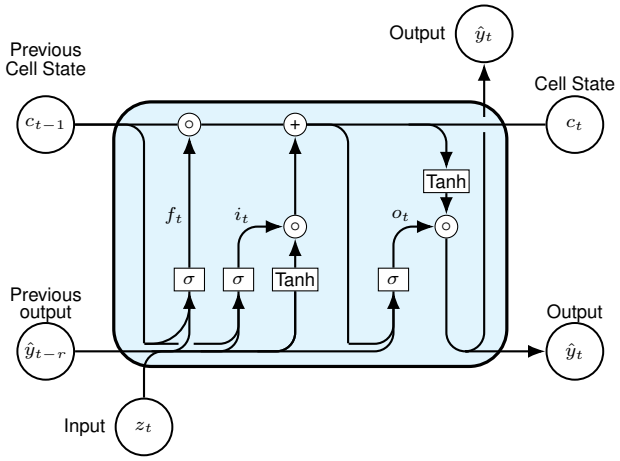

Fig. 3. Diagram of the ConvLSTM layer. Matrices W and bias b are omitted to improve clarity.

and

$$o_t = \sigma(W_{ZO} * z_t + W_{HO} * \hat{y}_{t-1} + W_{CO} \circ c_{t-1} + b_O)\,. \quad (2)$$

There are another two important tools in this layer: the *cell state* and the *forget gate*. The former keeps track of the information processed in previous time instants and the latter controls how the *cell state* is updated. The *cell state* is computed as

$$c_t = f_t \circ c_{t-1} + i_t \circ \tanh(W_{ZC} * z_t + W_{HC} * \hat{y}_{t-1} + b_C) \quad (3)$$

and the *forget gate* as

$$f_t = \sigma(W_{ZF} * z_t + W_{HF} * \hat{y}_{t-1} + W_{CF} \circ c_{t-1} + b_F)\,. \quad (4)$$

In these equations, σ symbolizes the sigmoid function and b_I, b_O and b_F represent a *bias* term for the respective state. The W terms represent the *weight matrices*, *e.g.* W_{Zi} is the weight matrix that connects the *input feature map* z_t to the *input gate*.

Finally, the output of this layer $\hat{y}_t$ is computed as

$$\hat{y}_t = o_t \circ \tanh(c_{t-1})\,. \quad (5)$$

In these expressions, $*$ represents the convolution operator and $\circ$ is the element-wise multiplication. Fig. 3 illustrates the operations that are described in equations 1 to 5.

Both the input and output of this layer are 4D tensors, where two dimensions represent space, a third corresponds to time and the forth represents the number of channels. In our proposed network, we use the output of the ConvLSTM directly as our estimate $\hat{y}_t$. As depicted on Fig. 4, the input z_t of the ConvLSTM layer at instant t, has the same resolution and number of frames of the output $\hat{y}_t$, differing only in the number of channels. This input of the ConvLSTM layer is composed of features computed by purely convolutional layers at different depths, as presented on the next subsection, resulting in a sequence of images with four channels. The output is a sequence of single-channel images, one in each time instant.

TABLE I
DETAILS OF THE LAYERS THAT WERE NOT IMPORTED FROM VGG16. THE MENTIONED CONV2D LAYERS CORRESPOND ONLY TO THOSE THAT WERE NOT IMPORTED FROM VGG16 (OUTSIDE YELLOW RECTANGLES IN FIGURE 4).

Layer type	Kernel size	Stride
ConvLSTM	3×3	1×1
Conv2D	1×1	1×1

C. Overall Architecture

The network's architecture[1] was chosen to easily assess the impact of learning temporal features in detection. With this goal in mind, we decided to use a popular neural network (VGG16 [38]) as a feature extractor. This choice intents to make performance comparison easier by using the feature extractor pre-trained on ImageNet thus accelerating training times. We have used features computed at different levels of the network, in a similar fashion to works like Ronneberger *et al.* [39]. In order to use features from the different levels, we need to adjust their resolution, which is done using an upsampling layer that performs nearest neighbour interpolation. We also reduce their number of channels before concatenation, using 2D convolutional layers. The details of these convolutional layers as well as the ConvLSTM are presented in Table I. In our case, the ConvLSTM is the last layer in the pipeline. Due to this fact, the output of ConvLSTM that in many works is usually denoted as hidden state happens to be the output and therefore is the predicted map $\hat{Y}_t$.

As already mentioned, this network's purpose was to evaluate the influence of temporal features, therefore it is intentionally simple. While we use only one ConvLSTM layer, more could be added. In particular, the feature extraction section, which now is carried out by pre-trained VGG16, might be replaced by recurrent layers. In [37], the authors obtained better results using deeper neural networks, composed only of ConvLSTM layers but, in practice, we found those configurations harder to train due to memory limitations and training speed.

D. Introducing Domain-Specific Knowledge

In maritime monitoring missions, we know the flight altitude and the camera parameters, therefore we have estimates of the area being observed. Additionally, we also have estimates of the maximum size of boats and the number of boats in the image at a given time. In our method, we use this information to guide the training process by incorporating this domain-specific knowledge into the loss function. The loss function is composed of two parts; we name the domain-specific part as area loss and depends on the average area labeled as containing a boat. The average area is written as

$$\bar{A}_{boat}(\hat{Y}_t) = \frac{1}{K} \sum_{k=0}^{K} \sum_{m=1}^{M} \hat{y}^m_{t-(k \times r)} \quad (6)$$

where y^m_t is the m^{th} pixel of prediction map y_t, M is the number of pixels in each image and K is the number of images

[1]The implementation of the architecture described in this section is included in the following repository: https://bitbucket.org/gccruz/convlstm_detection.git

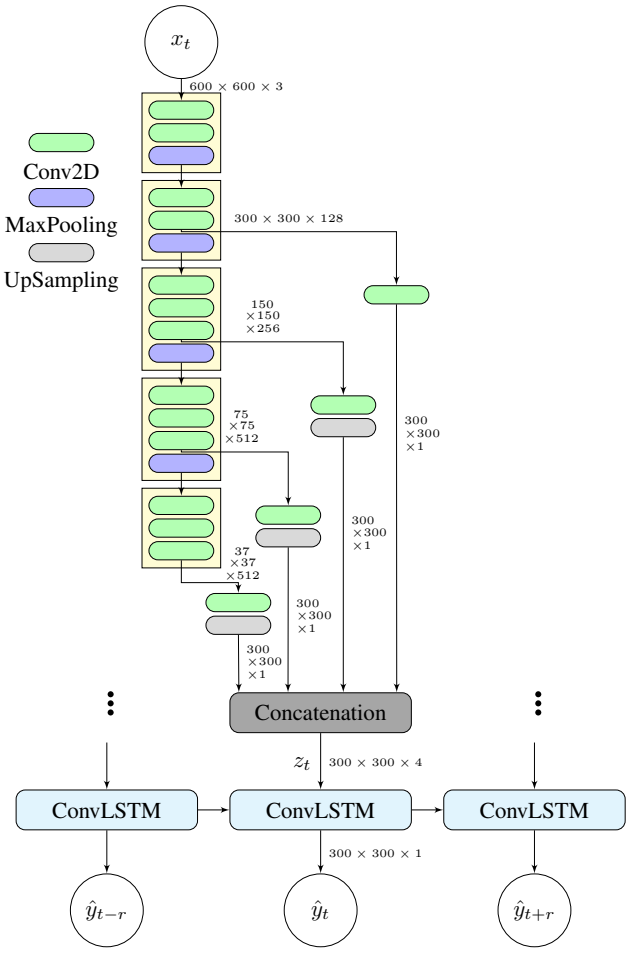

Fig. 4. Full network architecture. Enclosed in yellow rectangles are the pre-trained VGG layers. The features computed at different levels of VGG layers are then resized to match ConvLSTM's input size. As presented in Subsection III-B, the ConvLSTM not only receives the features computed in a given time instant but also the cell and hidden state from the previous time instant (best viewed in color).

in each sequence. The area loss, that depends on the average area, is written as

$$\mathcal{L}_{\text{area}}(\hat{Y}_t) = \begin{cases} 0, & \text{if } \bar{A}_{boat}(\hat{Y}_t) < s \\ \bar{A}_{boat}(\hat{Y}_t), & \text{otherwise} \end{cases} . \tag{7}$$

The behavior of area loss, shown in Fig. 5, favors the network to predict $\hat{Y}_t$ with a limited amount of pixels labeled as *one*. This prevents the network from considering large areas as boats, which normally is caused by the presence of glare. In our application, when computing the area loss $\mathcal{L}_{\text{area}}$, we considered $s = 5000$, which is a very permissive value. If we consider an unfavorable case, like depicted on Fig. 6, using our camera with smallest field of view (approximately $90°$) pointing directly downwards, mounted on an aircraft flying at 100 meters above the sea (lower than usual), the predicted map will have a maximum spatial resolution of approximately $0.66\,\mathrm{m/px}$. With this spatial resolution, one vessel with length of $120\,\mathrm{m}$ and beam of $20\,\mathrm{m}$ will occupy at most 4000 pixels

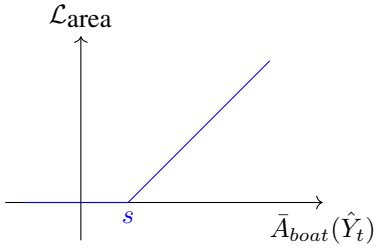

Fig. 5. Loss function L_{area}.

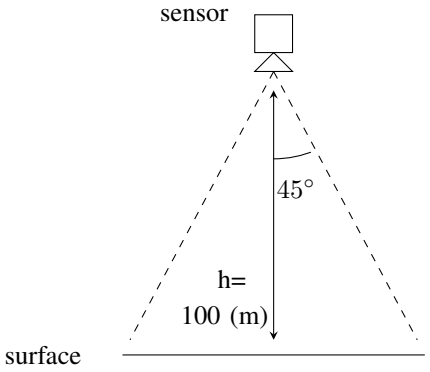

Fig. 6. Diagram of one unfavorable condition for the loss area, where a given boat might be closer to the sensor mounted on the aircraft.

in the map y_t. If each of those pixels is correctly labeled with value *one* and assuming the boat is visible throughout the sequence, then $\bar{A}_{boat}(\hat{Y}_t) = 4000$ and there is no penalization.

The second part of the loss is the binary cross-entropy and is defined as

$$\mathcal{L}_{c.e.}(Y_t, \hat{Y}_t) = -\frac{1}{MK}\sum_{k=0}^{K}\sum_{m=1}^{M}\Big[y^m_{t-(k\times r)}\log(\hat{y}^m_{t-(k\times r)}) + (1-y^m_{t-(k\times r)})\log(1-\hat{y}^m_{t-(k\times r)})\Big] . \tag{8}$$

With the two presented losses, area loss and binary cross-entropy, we compose the total loss as a weighted sum, as follows

$$\mathcal{L}_{total} = \mathcal{L}_{\text{c.e.}}(Y_t, \hat{Y}_t) + \lambda\, \mathcal{L}_{\text{area}}(\hat{Y}_t) \tag{9}$$

with $\lambda = 0.01$.

IV. Experiments

In the current section, we will present the experimental procedures, exhibit the obtained results and discuss them. In the next subsection, we will detail the characteristics of the dataset that was used to train and to test the detection methods. In the second subsection, we will study the configuration of time parameters (time span and time stride) that produced best results. On the third subsection, we will compare that configuration with four variations of the network: a network that did not used domain-specific knowledge for training, two with ConvLSTM replaced by a traditional LSTM and by a GRU and another with ConvLSTM replaced by a convolutional

layer. The goal of that test is to show the advantages of convolutional LSTM in this problem. The fourth subsection, will compare our approach with other detectors, in particular YOLO, a general purpose detector, and *detectnet+MHT*, which is a detector developed for maritime surveillance scenarios. In the last subsection, we expand our experiments, by using a different dataset and by studying the computational performance of our detector.

A. Dataset Description

We have used a subset of SEAGULL dataset [14]. The choice of this dataset is directly connected to the envisioned application of the detectors on board small fixed-wing UAVs. This type of application allows a good compromise of cost and complexity of sensors and the area that is surveyed.

SEAGULL dataset contains data from different spectra, namely visible light (RGB), Near Infrared, Long Wave Infrared and hyperspectral. We have chosen to consider only RGB videos since these are challenging and also because there is more data to train and test the different algorithms. From the RGB videos, we have selected sequences with different characteristics among them, to be more representative of a real-world application.

For training, we have used *seq07*, *seq08*, *seq09*, *seq12*, *seq14* and *seq15*. The first reason to choose these videos was the presence of challenging conditions. The second reason was their diversity, with different altitudes, different types of vessels and backgrounds.

For the test videos, we wanted to keep the same sequences (*seq02*, *seq06*, *seq13* and *seq16*) as presented in [18] for comparison purposes and we have included a highly demanding sequence: *seq17*. Because each of the sequences selected for testing depicts a different scenario, we will identify them using labels to have an easier correspondence (these were already used in [18]). The first video was named SUSP as two boats approach in a suspicious way; the second was designated SAR since a search and rescue raft was deployed; the third and fourth were labeled WIDE and NEAR, with a wide area being covered and with a boat being followed closely, respectively. The additional video sequence, *seq17*, contains a fast moving rigid hull inflatable boat with a long wake, thus we designated it as WAKE.

In Table II, we summarize some of the most relevant characteristics of the sequences, present the labels for the test videos and the main attributes of the videos for the training stage.

TABLE II
VIDEO SEQUENCES' CHARACTERISTICS.

Stage	Name	Resolution	Annotations	Objects	Label / Attributes
Test	*seq02*	1024 ×768	19621	2	SUSP
	seq06	1920 ×1080	8426	2	SAR
	seq13	1920 ×1080	2044	2	WIDE
	seq16	1920 ×1080	1237	1	NEAR
	seq17	1920 ×1080	504	1	WAKE
Train	*seq07*	1920 ×1080	739	3	yacht; shoreline
	seq08	1920 ×1080	236	3	strong glare; shoreline; strong wake
	seq09	1920 ×1080	358	1	patrol boat occluded by glare
	seq12	1920 ×1080	1276	1	high altitude
	seq14	1920 ×1080	1008	2	low altitude; patrol boat
	seq15	1920 ×1080	941	1	low altitude; strong glare

B. Effect of the image sequence's characteristics

One of the main aspects that we want to explore in our work, is how using temporal features can improve detection on maritime monitoring scenarios, by using a convolutional LSTM layer. Due to implementation details, during training, the full network which includes a recurrent layer has to be unrolled to perform forward and backward passes. Because of this fact, we have to specify the length of the sequence, that we designate as **time span**. Additionally, because changes from frame to frame are relatively slow, we choose to save some computation and sample frames contained in a given time span, skipping some in each processed sequence. We denote the spacing for sub-sampling as **time stride**. These two parameters are represented in Fig. 2.

To explore the impact of these configurations, we will present results evaluating each trained model on the test dataset. We have decided to separate the evaluation in two. The first part is carried out on four video sequences that offer challenges like glare and waves but the boats are visible in most cases. The second part is done on a sequence affected by a very strong wake that changes the boat's appearance and even occludes it.

For each condition, we evaluate the pixel Error Rate (ER) between the predicted map and the ground truth map, to get the combination that produces best results for ConvLSTM. The metric, ER, is computed as the ratio of incorrectly labeled pixels - False Positives (TP) and False Negatives (TN) - over their total number (Positives and Negatives), *i.e.*,

$$\mathrm{ER} = \frac{(\#FP + \#FN)}{(\#P + \#N)} \,. \tag{10}$$

This metric expresses the number of pixels from the predicted map that were incorrectly classified (either false positives or false negatives).

Table III presents the ER for the four video sequences, while Table IV presents the results in the WAKE sequence. For an easier comparability of the parameters' effects and since the frame rate is constant in the video sequences, we express time durations as number of frames. For instance, considering that the frame rate is 25 frames per second, a time stride of 5 frames, corresponds to sampling frames separated by 0.2 seconds. While many other configurations have a comparable performance, there are a few cases with a severely degraded performance. The worst cases occur when the time stride is half of the time span. This means that the

TABLE III
ERROR RATE RESULTS OBTAINED IN **SUSP**, **SAR**, **WIDE** AND **NEAR** VIDEOS FOR DIFFERENT TIME SPAN AND TIME STRIDE CONFIGURATIONS. The results are presented in percentage and the top score is highlighted in bold (smaller is better).

	Test A	Time Span (frames)[2] 10	20	40	60
Time Stride (frames)[2]	2	0.17	0.14	- [3]	- [3]
	4	- [4]	0.16	0.24	0.17
	5	0.23	0.15	**0.13**	0.18
	10	- [5]	0.19	0.15	0.14
	20	- [5]	- [5]	0.26	0.15

TABLE IV
ERROR RATE RESULTS OBTAINED IN **WAKE** VIDEO FOR DIFFERENT TIME SPAN AND TIME STRIDE CONFIGURATIONS. The results are presented in percentage and the top score is highlighted in bold (smaller is better).

	Test B	Time Span (frames)[2] 10	20	40	60
Time Stride (frames) [2]	2	0.14	0.14	- [3]	- [3]
	4	- [4]	0.14	0.25	0.14
	5	0.14	0.14	**0.13**	0.16
	10	- [5]	0.19	0.14	0.16
	20	- [5]	- [5]	0.78	0.15

number of processed frames is very small and the recurrent part of the network does not effectively use the mechanisms to keep/discard information over time.

In both cases, the optimal configuration was using a time stride of 5 and a time span of 40 frames (which corresponds to 1.6s). While one might think that the longer the time span, the better the results, it appears that there is a compromise between the length of the sequence and the frames spacing (time stride).

[2]These values are presented in number of frames for better interpretability. Since the frame rate is constant, each amount of frames has an equivalent amount of time, measured in seconds.

[3] This configuration resulted in using a significant amount of time instants, consequently it was very computationally demanding during training and therefore we decided not to use it.

[4] Since the Time Span is not divisible by the Time Stride, this configuration was discarded.

[5] This configuration resulted in too few time instants, thus we decided not to use it.

TABLE V
ERROR RATE RESULTS OBTAINED FOR OUR PROPOSED METHOD (CONVLSTM), A TRADITIONAL LSTM AND A PURELY CONVOLUTIONAL NETWORK. The first set of results was obtained using videos SUSP, SAR, WIDE and NEAR, and the second set was using video WAKE. The results are presented in percentage and the top score for each test is highlighted in bold (smaller is better)

Method	**SUSP**, **SAR**, **WIDE** and **NEAR** (%)	**WAKE** (%)
ConvLSTM with domain-specific knowledge	**0.13**	**0.13**
ConvLSTM without domain-specific knowledge	0.13	0.14
LSTM	0.22	0.22
GRU	0.17	0.17
ConvNet	0.15	0.20

C. Comparison with LSTM, GRU and convolutional network

After finding the best setting using a ConvLSTM trained with domain-specific knowledge, we will then compare our method with a ConvLSTM trained without domain-specific knowledge, a ConvNet, a GRU and traditional LSTM. The results presented in Table V highlight the benefit of using domain-specific knowledge at training time, the benefit of learning temporal features over processing each frame independently and also the benefit of using a ConvLSTM layer over the traditional LSTM and the GRU. The first comparison was obtained by training the network presented in Fig. 4 using the loss presented in Eq. 9 (using domain-specific knowledge) and by training using only binary cross-entropy (without using domain-specific knwoledge). The second benefit is demonstrated by using the architecture presented in Fig. 4 but replacing the ConvLSTM with a 2D convolutional (Conv2D) layer. This replacement makes the processing of each image independent of previous time instants, *i.e.* removes the connection between time instants. The third advantage is demonstrated by replacing the ConvLSTM layer, in Fig. 4, with a traditional LSTM and with GRU layer. When comparing with the LSTM and with GRU, we could not keep exactly the same network configuration due to the very large number of parameters. The output's size was changed to 30×30 pixels and the recurrent layers' input size had a resolution of 75×75 (which was transformed into a one column vector). Even with these adaptations, the networks using LSTM and GRU had approximately 98 and 77 million trainable parameters, respectively, opposed to ConvLSTM that had 14 million parameters (with an input and output size of 600×600 and 300×300 pixels).

In Table V, we present a test with two sets of images. In the first, we used videos SUSP, SAR, WIDE and NEAR, and on the second, we used video WAKE. Despite different resources' demand, the approach that produced worst results was LSTM. This was probably caused by the very large number of parameters to tune and the small resolution of both input X_t and output $\hat{Y}_t$. In particular, the low resolution forced us to approximate the ground truth Y_t when training and causing significant approximation errors.

The methods with intermediate performance were ConvNet and GRU. ConvNet obtained a score similar to the best approach in the test with four videos but had a worse performance on WAKE. This is caused primarily by the predominance of the wake. While on the test with four videos there are some challenges, like glare and different scale of the objects, the approach which is based solely on visual features can still produce interesting results. On the test with WAKE, because the appearance is so heavily affected by the wake, the visual features alone are not enough to achieve a good performance. GRU had a slightly worst performance than ConvNet on the set of four videos but, by learning temporal feautures, performed better on WAKE. It is also worth noting that GRU performed better than LSTM, which is consistent with previous results like presented by Jozefowicz *et al.* [40]. If the training data was more abundant, the LSTM could be abble to generalize better and the difference between the two

become smaller. When comparing GRU with ConvLSTM, the latter performed better in both tests. One of the main causes is the relatively reduced number of parameters allowed by the convolutional structure, which makes training easier.

Our method, the network with a ConvLSTM layer, trained on sequences with a time span of 40 frames and time stride of 5 frames, uses learned visual and temporal features, which have allowed it to obtain the best performance in both tests. It is important to note that the network that was trained using domain-specific knowledge achieved a slightly better result in the case of the video with strong wake. Despite being a small gain, it is valuable to verify that using a loss penalizing the prediction of large areas led better performance on discarding persistant phenomena like wakes. The network configuration using ConvLSTM and trained using domain-specific knowledge is used on the next subsection to compare with other already published detectors and demonstrate that learning temporal features produces benefits over those approaches.

D. Comparison with other detectors in SEAGULL dataset

The benchmarks that we used were a standard detection neural network (YOLO 9000 [19]) and a neural network associated with a Multiple Hypothesis Tracker - *detectnet+MHT* [18]. Both alternatives were retrained on the same dataset as our method.

YOLO, the standard neural network is not the top scoring detector in the literature but presents one of the best compromises between speed and performance, therefore becomes an adequate candidate to process a stream in real time. This network is composed of convolutional layers that predict a grid, where each cell may have multiple bounding boxes, each with its coordinates, confidence score and class probability.

The second method uses a detection neural network simpler than YOLO but explores time coherence among detections in consecutive instants. The network creates a grid that indicates if an object of interest is present on each cell and computes a regression for the location of a bounding box. The bounding boxes created at each time instant are then used to create a level of a graph, where each node corresponds to a detection, weighted by the detection score. Nodes of consecutive levels are connected by edges, which are weighted based on the Euclidean distance between bounding boxes. MHT then computes the probability of a given tracklet to correspond to a correct detection by searching combinations of nodes and corresponding edges with high scores.

The output of both compared methods is bounding boxes, therefore we adapted our method to get the same type of output. Starting with the prediction map, Y_t, that was already mentioned, we obtain binary maps by thresholding Y_t. Afterwards, we compute the bounding box for each of the blobs presented in the binary image. This technique is not as advanced as the regression layers used in the compared methods but allows us to evaluate the performance without adding layers that needed to be trained with the rest of the network and might conceal the behavior of the ConvLSTM layer.

Unlike the previous evaluation, to compare the method presented in this work with other detectors, we did not use binary cross-entropy but used an evaluation more common for detectors. We have used the evaluation metric that was adopted in [18], to keep the same scoring process. This metric itself was adapted from Dollár *et al.* [41]. To validate a detection, the mentioned authors required an intersection over union (IoU) bigger than 50% and defined it as

$$\text{IOU} = \frac{\text{area}\{\bar{D}^t \cap G^t\}}{\text{area}\{\bar{D}^t \cup G^t\}}, \tag{11}$$

where D^t and G^t denoted the detections and the ground truth. While this is adequate for many applications, just as in [18], we believe that in the present scenario, 10% is enough. Given this matching criterion, two quantities are computed: Precision and Recall. These are respectively defined as *Precision* = # true positive (TP)/ (# TP + # false positive (FP)) and *Recall* = # TP/ (# TP + # false negative (FN))[6]. With Precision and Recall, we have obtained the results presented in Fig. 7. These plots show the behavior of detectors through their operating range but it is useful to have a quantity to summarize and compare the complete range. We selected Area Under the Curve (AUC), which is computed as the sum of Precision $p(n)$, at every possible threshold with the index n, times Recall's variation $\Delta r(n)$ between these points, *i.e.*,

$$AUC = \sum_{n=1}^{N} p(n)\Delta r(n). \tag{12}$$

The AUC obtained with each method in each video is presented in the corresponding legend.

When inspecting the results, it is worth noting that, independently of the method, there is a substantial difference in performance between NEAR, WIDE and the rest of the sequences. In the mentioned sequences there are some challenges but the appearance of the boats is more or less constant. The three other sequences have much more challenging conditions, like two boats close to each other that are detected as one, a very small life raft and a wake that cloaks the boat

From the first four plots, we can verify that the behavior of the three methods is similar. Typically *detectnet+MHT* has a high precision but achieves a smaller recall. This is caused by the MHT which discards many false positives but when more demanding conditions occur (like abrupt movement), it also discards true positives. YOLO has a smoother decrease in Precision as Recall increases. Our proposed method shows a comparable performance and the AUC is similar in the first three videos. In the fourth video (NEAR), ConvLSTM falls short of the other two approaches. This inferior performance of the proposed method, on the third video, is not caused by lack of sensitivity of the detector but rather by inadequate size of the bounding boxes. In several instants, the size's difference of the ConvLSTM detection box, shown in red in Fig. 8(d) and the object, leads to false negatives, *i.e.* the IoU

[6] False positives and false negatives in this paragraph refers to incorrect bounding boxes and to missing bounding boxes, respectively. This differs from the false positives and false negatives that were used to compute the Error Rate mentioned before.

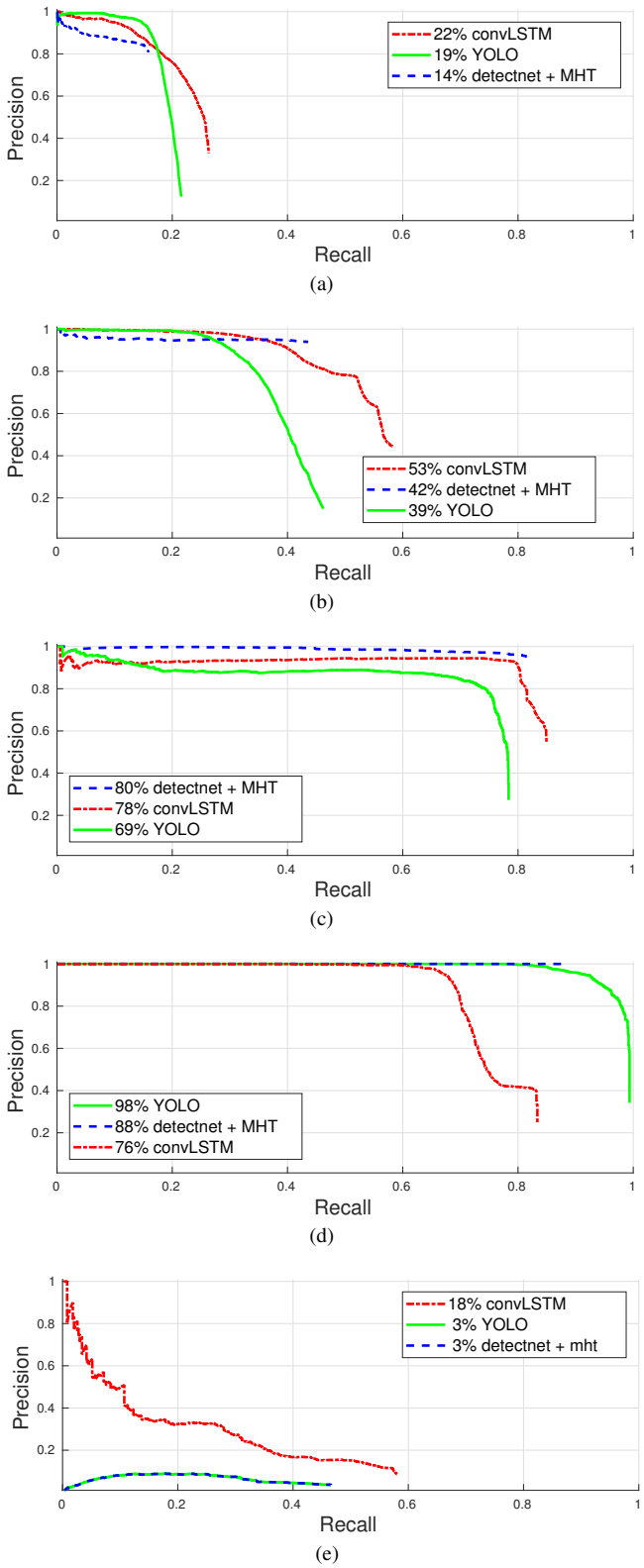

Fig. 7. Results of evaluation using a traditional detection metric [41], with an overlap threshold of 10%. Results were obtained for sequence (a) SUSP, (b) SAR, (c) WIDE, (d) NEAR and (e) WAKE.

is lower than required and no detection matches the ground truth. Another relevant factor is that the boat enters and leaves the field of view frequently. Since ConvLSTM learns temporal features, it tends to mark persistent objects as detections and to discard objects with limited duration (like waves) which tipically correspond to noise. Thus, when the boat enters the field of view, our method does not imediatly marks it as a detection, it requires some time instants to do so. The decrease in Recall, caused by the boat leaving and entering the image, also affects *detectnet+MHT*, which also needs several time instants with the object of interest in the field of view, to consider a valid detection.

On the last video sequence, the boat is moving at high speed causing a wake several times bigger than the vessel. This condition affects the performance dramatically with both *detectnet+MHT* and YOLO having a very small AUC. Our method is also affected significantly but still manages to have an AUC order of magnitude higher than the other methods. This happens because other methods rely heavily on learning the appearance of boats from the training dataset. Even *detectnet+MHT*, which verifies time consistency, does not use this information to create detections but only to validate detections based on appearance. Our method, on the other hand, learns not only the appearance but also the motion model and uses these two features to create detections.

E. Additional experiments

In order to have a more comprehensive evaluation of our approach, we performed tests on a different dataset and also measured the computational performance of the different methods. The goal of the experiment in a different dataset is to verify if the detectors are able to generalize or if they overfit to SEAGULL dataset. The evaluation of the detectors' computational performance, assesses if it is possible to use them in a real world application.

1) Testing on MARDCT dataset: The dataset that was chosen for these additional tests was MARDCT [11]. This dataset was gathered by the ARGOS system that monitors the Venice Grand Canal. ARGOS' cameras are installed in buildings, consequently most objects of interest are very close to the camera and some urban elements, like walls, are present. Due to this fact, we had to carefully select videos with some similarity to our scenario. The elected videos were *wake-1*, *wake-2* and *wake-3*. The properties of the three videos differ from SEAGULL dataset: the resolution is smaller, the line of sight from the camera to the objects of interest was almost paralel to the water surface and the type of boats is also different. The apparent movement of the image is also distinct, with long periods with movement caused only by the boats and short periods with pan movement, causing severe blur.

It is important to note that, in this case, there is no distance's information from the camera to the boats, hence there is no guarantee that the Domain-Specific Knowledge included at training time is beneficial. Despite this shortcoming, we applied the same three detectors that were already used in the previous subsection without retraining. For brevity, we condensed the results as AUC in Table VI.

(a)

(b)

(c)

(d)

(e)

Fig. 8. Examples of the bounding boxes obtained with the three methods in each of the sequences. The bounding boxes are colored according to the method: YOLO is represented in green, *detectnet+MHT* in blue and the proposed method (based on ConvLSTM) is represented in red. Images on the left correspond to original images from video sequences: (a) SUSP, (b) SAR, (c) WIDE, (d) NEAR and (e) WAKE. Images on the right present the detections in greater detail.

TABLE VI
AREA UNDER THE CURVE (INDICATED IN %) OBTAINED WITH *detectnet+MHT*, YOLO AND CONVLSTM, IN VIDEO SEQUENCES FROM MARDCT DATASET. THE EVALUATION METHODOLOGY [41] WAS THE SAME AS IN FIG. 7.

Detector	Video sequences		
	wakes-1	*wakes-2*	*wakes-3*
detectnet+MHT	0	58	34
YOLO	8	97	4
ConvLSTM	11	77	62

As shown in the Table VI, the video in which the detectors performed worst was *wakes-1*. The main causes are the very low resolution (240×320) of the original video making objects pixelated and also the presence of haze, as shown in the right side of Fig. 9(a), which hinders the visibility of some boats. In this video, *detectnet+MHT* does not generate any valid detection. On the other two videos, *detectnet+MHT* achieves modest scores, which are caused by the low resolution (704×576) and also by the boats repeatedly entering and leaving the image. Many parts of the third video are unfocused and *detectnet+MHT* tipically fails to detect on those ocasions. YOLO, on the other hand, obtains a very high score on *wakes-2* and a low score on *wakes-3*. This occurrence is strongly influenced not only by unfocused images but also by the presence of mountains on the horizon, leading to many false detections on that area, as presented on Fig. 9(c). Our method also struggles with the conditions present in the first video, scoring just slightly above YOLO. On the other two videos, it obtains higher Areas Under the Curves and the phenomena that degrades its performance the most, is the presence of wake detached from the boat. As mentioned before, the three methods were applied to MARDCT without retraining, so the disparity in performance of YOLO might indicate that it has overfitted to a given appearance of boats. The ConvLSTM method, on the other hand, had more consistent results, which show a greater generalization capability.

2) Computational performance: To understand the applicability of our method to a real world scenario, we gauged the execution speed of the different methods and the number of parameters used by the neural networks in each method. For the considered application (maritime monitoring using aerial vehicles), the execution speed strongly conditions the applicability, however, it depends immensely on the hardware, libraries and optimizations that are used. Thus, our goal is primarily to compare our method with YOLO, one of the approaches with best compromises between accuracy and speed, and not to have an absolute measurement. The execution times, presented in Table VII, were obtained using a Keras [42] implementation with Tensorflow backend, running on Nvidia Titan XP GPU. The number of parameters also affects the applicability, in particular if embedded applications are considered. Embedded applications tipically use hardware with less memory, which may inhibit the use of models with a large number of parameters.

As shown in Table VII, YOLO is almost ten times faster than the other approaches. This difference is easily explained by the fact that the method based on ConvLSTM needs to ex-

(a) (b) (c)

Fig. 9. Examples of the bounding boxes obtained with the three methods in: (a) *wakes-1*, (b) *wakes-2* and (c) *wakes-3*. The bounding boxes are colored according to the method: YOLO is represented in green, *detectnet+MHT* in blue and ConvLSTM in red.

TABLE VII
ASSESSMENT OF COMPUTATIONAL PERFORMANCE OF THE DIFFERENT METHODS.

Method	Average execution time (s)	Number of Parameters (Approx.)
detectnet+MHT	0.25	6×10^6
YOLO	0.03	50×10^6
ConvLSTM	0.19	14×10^6

tract features from several images before feeding the information into the recurrent layer (since we have used, respectively, a time span and a time stride of 40 and 5, we process 8 images). When considering the number of parameters, YOLO is the most demanding option and *detectnet+MTH* is the lightest alternative. Our method obtains an intermediate value, with three times less parameters than YOLO. Since nowadays there are several embeded applications of YOLO, even in FPGAs [43], there are also good prospects to sucessfully implement our method on the same type of platforms.

V. CONCLUSIONS

This paper presents a method to learn not only spatial features but also temporal features present in video sequences. The usage of temporal features attempts to improve the detection of maritime objects in video sequences, which contain strong distractors like glare and wakes. This method is composed of two main parts, one spatial feature extractor based on VGG network and one recurrent layer, the Convolutional LSTM. The latter is the key component to learn temporal features since it has a memory cell that keeps or forgets information, according to the situation. Unlike traditional LSTMs, some operations in this layer are applied convolutionally which removes a significant spatial redundancy.

This method is evaluated with two kinds of tests. The first test investigates what is the configuration (number of frames, length and time stride) that produces best binary maps representing the position of boats and then compares the proposed approach with traditional LSTM and with the purely convolutional network (ConvNet). The comparison shows that there is a performance gain of the proposed approach over the other two. The second test evaluates the quality of generated bounding boxes against two detectors. The performance of the three methods is comparable in four videos out of five. The fifth video, however, is very challenging and our proposed method achieves a score several times higher.

Given the obtained results (especially with the second test), we conclude that learning temporal features is useful for maritime detection in videos captured by small aircraft. In the future, we would like to explore more configurations, in particular stack more recurrent layers and also extend the time span considered by ConvLSTM. As shown in the SEAGULL dataset, some videos contain objects of interest with faint visual features and the temporal features learned by ConvLSTM can improve the knowledge about a given scenario. However, for conditions where the object of interest is clearly visible and especially when its size is larger, other detectors can generate better detections.

With these considerations in mind, a real-world application could benefit from using a combination of detectors that might be selected according to the mission or scenario. Another path that we would like to investigate in the future is the use of contextual information available on-board like altitude, aircraft's attittude and sensor's parameters to improve detection. One possibility, is the use of these parameters to create an additional input channel, where each pixel contains the slant range from the sensor to the observed area. The range information would prevent detections with large areas in regions that are very far or detections with very small areas in regions that are very close.

ACKNOWLEDGMENTS

This work was partially supported by FCT project VOAMAIS (02/SAICT/2017/31172). The authors would like to thank all the VisLab and Portuguese Air Force Research Center team that allowed the collection and labeling of the video sequences.

REFERENCES

[1] I. M. Association *et al.*, "International shipping facts and figures–information resources on trade, safety, security, and the environment," *London: International Maritime Association*, 2011.

[2] D. Hinrichsen, "The coastal population explosion," in *Trends and Future Challenges for US National Ocean and Coastal Policy: Proceedings of a Workshop*, vol. 22. NOAA, January 22, 1999, Washington, DC, 1999, pp. 27–29.

[3] "Piracy and armed robbery against ships," ICC International Maritime Bureau, Tech. Rep., 2016.

[4] I. O. f. M. IOM, "Mediterranean update migrant deaths rise to 3,329 in 2015," October 2015. [Online]. Available: https://www.iom.int/news/mediterranean-update-migrant-deaths-rise-3329-2015

[5] H. K. White, P.-Y. Hsing, W. Cho, T. M. Shank, E. E. Cordes, A. M. Quattrini, R. K. Nelson, R. Camilli, A. W. Demopoulos, C. R. German *et al.*, "Impact of the deepwater horizon oil spill on a deep-water coral community in the gulf of mexico," *Proceedings of the National Academy of Sciences*, vol. 109, no. 50, pp. 20 303–20 308, 2012.

[6] M. Fingas and C. Brown, "Review of ship detection from airborne platforms," *Canadian journal of remote sensing*, vol. 27, no. 4, pp. 379–385, 2001.

[7] J. N. Briggs, *Target detection by marine radar*. IET, 2004, vol. 16.

[8] G. Fein and M. J. I. D. Review, "Sentient's vidar shows success during uscg scaneagle demonstrations," Apr 2018. [Online]. Available: https://www.janes.com/article/79300/sentient-s-vidar-shows-success-during-uscg-scaneagle-demonstrations

[9] L. Zabala, 2018. [Online]. Available: https://www.nbcsandiego.com/on-air/as-seen-on/Drone-Helps-Coast-Guard-Intercept-Drug-Shipments_San-Diego-471150933.html

[10] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[11] D. D. Bloisi, L. Iocchi, A. Pennisi, and L. Tombolini, "Argos-venice boat classification," in *Advanced Video and Signal Based Surveillance (AVSS), 2015 12th IEEE International Conference on*. IEEE, 2015, pp. 1–6.

[12] X. Bao, S. Zinger, R. Wijnhoven *et al.*, "Robust moving ship detection using context-based motion analysis and occlusion handling," in *Sixth International Conference on Machine Vision (ICMV 13)*. International Society for Optics and Photonics, 2013, pp. 90 670F–90 670F.

[13] Y. Bazi and F. Melgani, "Convolutional svm networks for object detection in uav imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 6, pp. 3107–3118, 2018.

[14] R. Ribeiro, G. Cruz, J. Matos, and A. Bernardino, "A dataset for airborne maritime surveillance environments," *IEEE Transactions on Circuits and Systems for Video Technology*, 2017.

[15] "Airbus maritime dataset," https://sandbox.intelligence-airbusds.com/web/, accessed: 2018-08-22.

[16] J. S. Marques, A. Bernardino, G. Cruz, and M. Bento, "An algorithm for the detection of vessels in aerial images," in *Advanced Video and Signal Based Surveillance (AVSS), 2014 11th IEEE International Conference on*. IEEE, 2014, pp. 295–300.

[17] G. Cruz and A. Bernardino, "Aerial detection in maritime scenarios using convolutional neural networks," in *International Conference on Advanced Concepts for Intelligent Vision Systems*. Springer, 2016, pp. 373–384.

[18] ——, "Evaluating aerial vessel detector in multiple maritime surveillance scenarios," in *OCEANS Anchorage, 2017*. IEEE, 2017, pp. 1–9.

[19] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," *arXiv preprint*, 2017.

[20] P. Liang, H. Ling, E. Blasch, G. Seetharaman, D. Shen, and G. Chen, "Vehicle detection in wide area aerial surveillance using temporal context," in *Information Fusion (FUSION), 2013 16th International Conference on*. IEEE, 2013, pp. 181–188.

[21] X. Shi, H. Ling, E. Blasch, and W. Hu, "Context-driven moving vehicle detection in wide area motion imagery," in *Pattern Recognition (ICPR), 2012 21st International Conference on*. IEEE, 2012, pp. 2512–2515.

[22] A. Gaszczak, T. P. Breckon, and J. Han, "Real-time people and vehicle detection from uav imagery," in *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics, 2011, pp. 78 780B–78 780B.

[23] M. Teutsch and W. Kruger, "Robust and fast detection of moving vehicles in aerial videos using sliding windows," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2015, pp. 26–34.

[24] Z. Shi, X. Yu, Z. Jiang, and B. Li, "Ship detection in high-resolution optical imagery based on anomaly detector and local shape feature," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 8, pp. 4511–4523, 2014.

[25] M. Dawkins, Z. Sun, A. Basharat, A. Perera, and A. Hoogs, "Tracking nautical objects in real-time via layered saliency detection," in *SPIE Defense+ Security*. International Society for Optics and Photonics, 2014, pp. 908 903–908 903.

[26] T. Cane and J. Ferryman, "Saliency-based detection for maritime object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 18–25.

[27] A. Sobral, T. Bouwmans, and E.-h. ZahZah, "Double-constrained rpca based on saliency maps for foreground detection in automated maritime surveillance," in *Advanced Video and Signal Based Surveillance (AVSS), 2015 12th IEEE International Conference on*. IEEE, 2015, pp. 1–6.

[28] P. Westall, J. J. Ford, P. O'Shea, and S. Hrabar, "Evaluation of maritime vision techniques for aerial search of humans in maritime environments," in *Digital Image Computing: Techniques and Applications (DICTA), 2008*. IEEE, 2008, pp. 176–183.

[29] F. Maire, L. Mejias, and A. Hodgson, "A convolutional neural network for automatic analysis of aerial imagery," in *Digital Image Computing: Techniques and Applications (DICTA), 2014 International Conference on*. IEEE, 2014, pp. 1–8.

[30] F. Bousetouane and B. Morris, "Fast cnn surveillance pipeline for fine-grained vessel classification and detection in maritime scenarios," in *Advanced Video and Signal Based Surveillance (AVSS), 2016 13th IEEE International Conference on*. IEEE, 2016, pp. 242–248.

[31] L. W. Sommer, T. Schuchert, and J. Beyerer, "Fast deep vehicle detection in aerial images," in *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on*. IEEE, 2017, pp. 311–319.

[32] Q. Wang, Z. Yuan, Q. Du, and X. Li, "Getnet: A general end-to-end 2-d cnn framework for hyperspectral image change detection," *IEEE Transactions on Geoscience and Remote Sensing*, no. 99, pp. 1–11, 2018.

[33] S. Oh, S. Russell, and S. Sastry, "Markov chain monte carlo data association for multi-target tracking," *IEEE Transactions on Automatic Control*, vol. 54, no. 3, pp. 481–497, 2009.

[34] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg, "Multiple hypothesis tracking revisited," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4696–4704.

[35] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[36] Q. Wang, S. Liu, J. Chanussot, and X. Li, "Scene Classification With Recurrent Attention of VHR Remote Sensing Images," pp. 1–13, 2018.

[37] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," in *Advances in neural information processing systems*, 2015, pp. 802–810.

[38] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[39] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[40] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *International Conference on Machine Learning*, 2015, pp. 2342–2350.

[41] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 4, pp. 743–761, 2012.

[42] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[43] H. Nakahara, M. Shimoda, and S. Sato, "A demonstration of fpga-based you only look once version2 (yolov2)," in *2018 28th International Conference on Field Programmable Logic and Applications (FPL)*. IEEE, 2018, pp. 457–4571.

Gonçalo Cruz (M.Sc. 2012) received the M.Sc. degree in electrical and computer engineering from the Portuguese Air Force Academy, Sintra, Portugal and he is also a Ph.D. student at the Instituto Superior Técnico, Lisbon University, Portugal. He is a lecturer with the Portuguese Air Force Academy and a researcher with the Portuguese Air Force Research Center, working in the area of computer vision applied to unmanned aerial vehicles. His research interests include machine learning and computer vision applied to aerial robotics.

Alexandre Bernardino (Ph.D. 2004) is an Associate Professor at the Dept. of Electrical and Computer Engineering and Senior Researcher at the Computer and Robot Vision Laboratory of the Institute for Systems and Robotics at IST, the faculty of engineering of Lisbon University. He has participated in several national and international research projects as principal investigator and technical manager. He published more than 40 research papers in peer-reviewed journals and more than 100 papers on peer-reviewed conferences in the field of robotics, vision and cognitive systems. He is associate editor of the journal Frontiers in Robotics and AI and of major robotics conferences (ICRA, IROS). He has graduated 10 Ph.D. students and more than 40 M.Sc. students. He was co-supervisor of the Ph.D. Thesis that won the IBM Prize 2014 and the supervisor of the Best Robotics Portuguese MSc thesis award of 2012. He is the current chair or the IEEE Portugal Robotics and Automation Chapter. His main research interests focus on the application of computer vision, machine learning, cognitive science, and control theory to advanced robotics and automation systems.