SNAME
THE INTERNATIONAL COMMUNITY
FOR MARITIME AND OCEAN PROFESSIONALS

THE SOCIETY OF NAVAL ARCHITECTS AND MARINE ENGINEERS
TEXAS SECTION

VIRTUAL MULTIMODAL AUTOMATED OBJECT DETECTION WITH DEEP NEURAL NETWORKS

NIKOLAOS MITSAKOS
DEPARTMENT OF MATHEMATICS, UNIVERSITY OF HOUSTON

SANAT UPADHYAY
INNOVATION CENTER, UNIVERSITY OF HOUSTON AND LOLAARK LLC

MANOS PAPADAKIS
DEPARTMENT OF MATHEMATICS, UNIVERSITY OF HOUSTON

Proceedings of the 24th Offshore Symposium, February 20th 2019, Houston, Texas
Texas Section of the Society of Naval Architects and Marine Engineers

---

ABSTRACT

In this work, we attempt to illustrate possible applications of automated object detection in video sequences and still images using state-of-the-art artificial intelligence methods, such as convolutional neural networks. These novel tools can be used in various application domains in offshore operations. Neural networks can be trained to detect humans, specific objects and even learn how to conduct pipeline, anchor, tank and other types of inspections where they can significantly reduce the possibility of human error, not by replacing the human operator, but by helping him or her to be more effective and efficient. One of the problems we will discuss is how to harness the power of image pre-processing methods to increase the informative content of inputs, especially in environments where video or still-image visibility may be poor due to weather conditions, underwater turbidity or smoke.

Keywords: Water turbidity, fog, uneven illumination, visibility improvement, object detection, multimodal deep learning, convolutional neural networks, virtual lookout,

1 INTRODUCTION

Artificial intelligence for big data management is gradually becoming a reality of life, although there is a significant need for theoretical work in the understanding of how neural networks with a significant number of hidden layers work. They can be used for decision-making and harness the innate ability of neural networks to handle multiple inputs to generate one output, namely a decision or suggestion. Prior to the AI revolution, the trend to load airplane cockpits with numerous sensors resulted in a number of multiple inputs often in the form of audible alarms or visual displays simultaneously activated. This is one instance, where numerous information input channels have to be simultaneously interpreted by a single operator, namely the pilot, who has to make a decision explaining the causes of these alarms. Aviation psychologists even in the 80s have worked on this problem of the increased monitoring workload of humans and concluded, that big amount of information provided to pilots by monitoring systems and warnings poses as equal problem as limited amount of information would do Endsley (1999). Needless to mention that in aviation multiple alarms are often related to serious flight incidents.

Although not smarter than humans, AI can handle more effectively information from multiple sensors, but to do so, AI systems often require excessive training. We can think of an artificial intelligence system, as an integrated piece of software and hardware which will interpret a condition and assign to it a label, or an interpretation of a situation. This integrated system may function properly only if any situation it encounters fits well within the probabilistic parameters of its training. Anything outside its experience, will only trigger a random response. One way to increase its experience, especially in cases, which are not going to be fit properly within its knowledge space, is to use multiple inputs. One example is ABB's experimental Virtual Lookout, which combines visual camera inputs with LIDAR sensors too measure distance from various visible targets. Hyperspectral imaging is another example of the use of multiple sensors or modalities to help interpret information. This modality has been used for the detection of underwater pipeline corrosion, or from airborne sensors to detect types of marine life or water pollution, e.g. Roper & Dutta (2006), Johnsen et al. (2016). In the jungle of Deep Learning, *multimodal learning* has been used with audio and video with of lips movement for speech recognition Ngiam et al. (2011), Srivastava & Salakhutdinov (2012), and images combined with distance data obtained by RGB-D sensors for classification Eitel et al. (2015).

In this work we use a single input to generate artificially additional inputs, which we fuse, as we would do, had we have more than one physical modality. In no case, do we advocate that additional physical modalities are not useful. Definitely they are, but our purpose is to hint that virtual modalities can potentially help the automated decision process, when the input sensor operates in conditions which may potentially deliver less reliable inputs. To investigate the role of virtual modalities in improving the overall system accuracy, we use a top performer Neural Network, the Faster R-CNN. In fusing the virtual with the physical inputs, we chose not to change the network's basic elements of architecture.

AI acolytes may argue that when we train systems, we must include examples even from marginal conditions. However, there is always a risk of using an excessive number of training examples to account for a multitude of acquisition conditions which render the system unable to generalize, that is to be unlikely to derive a correct decision, when an input presented to it is not similar to the set of its experiences. The size of the training set depends not only on the number of labels the system must learn to output, but also on the inherent variability within the class of same label inputs and the variability of input acquisition conditions. Taking all three into account it is not difficult to realize that the training set quickly grows immensely.

The remedy we propose for this situation is to help the system improve its generalizability by reducing the set of conditions influencing the needed volume of training examples, and compensate for this reduction by adding the inputs from the virtual modalities. Moreover, the need to have a huge training set requires an equally big data collection enterprise, which may be both costly and time consuming. It is not hard to be trapped into this kind of predicament. For instance, if we want to develop a system to detect metal surface corrosion, we need to present to the system a balanced number of training examples from healthy and corroded surfaces. Here, we need to introduce the term *ground truth* which is widely used when referring to training examples. The key work here is the word "balanced", which requires a possibly time-consuming data collection from corroded surfaces with medium of low levels of deterioration. In such cases, the best approach is to collect data samples of what is designated as normal, and use a NN which scores samples presented to it. Score outputs express the network's belief that an examined surface patch is healthy. In this case, with a properly trained system we can reasonably expect that patches with signs of deterioration will be outliers and therefore will receive low scores, which are often called by the NN community as *confidence scores*.

We begin our presentation with a brief overview of the two virtual modalities we use, followed by a sketchy description of the neural network architecture we use to fuse the physical and virtual modalities inputs. Finally we close with our experimental results.

2 METHODOLOGY

2.1 Digital Image Enhancement and Illumination Neutralization

Digital image enhancement is the process of adjusting the pixel values of a digital image, with the purpose of deriving an image which is visually more appealing and/or more informative. To this end, a broad range of image enhancement techniques have been proposed, starting from the simplest brightness, contrast or sharpness adjustment, to the removal of various types of distortion, such as noise created by the digital camera sensor. In addition, to this distortion type, image enhancement methods, often, focus on specifically enhancing specific image informative content, such as edges (e.g. Canny Edge Detector) or texture. However, there are image acquisition conditions which inhibit the capture of a still image or video footage with enough information to allow these image enhancement methods to fulfill their role. Examples of such conditions are low-light, especially at night with insufficient illumination, shadows cast by objects, and the presence of light scatterers, such as haze, fog, water vapors in general, and underwater suspended particles. Object detection by artificially intelligent systems depends on the extraction of features, or more simply, the generation of identifying signatures which indicate the presence of a certain local shape in an image indicative of the presence of a certain object, e.g. a cylindrical surface. As more objects may share the same shape, e.g. in our example both cans and pipes have this shape, the AI system will learn how to piece together these signatures and conclude that *a certain type of an object is present at a certain location of an image* and perhaps quantify its belief on how reliable the detection is. This is a relatively complex method. A system under supervised learning achieves this goal to a certain extend. In the next section we discuss this issue in greater detail.

State of the art object detectors are exposed through supervised training to a variety of imaging conditions. so low illumination, or shadows do not often pose significant problems to them. But when imaging conditions, such as water vapors, haze, smoke or water turbidity, in general conditions conducive to the higher than normal scattering of light rays, detections become acutely uncertain. So, what if there are methods which can essentially "lift the veil of invisibility" and make the object detector to work properly? To make this point more clear, we stress that the extra variability these exterior conditions impose create perturbations to a classifier which do not necessarily fit well within its learned experiences. Consequently, ameliorating their effects on an input image makes the objects in the input more similar to the classifier's experience. In this work we used two such methods, originally designed to neutralize the effects of uneven or low illumination. We use these methods to create supplementary features, as if we incorporate them as separate inputs from additional modalities.

Illumination neutralization algorithms into two main categories Ochoa-Villegas et al. (2015): Those that treat the illumination problem as a *relighting* problem and those that treat it as an *unlighting* one. Relighting methods improve similarity of illumination conditions between gallery images and a probe image Unlighting methods utilize the Lambertian reflectivity principle:

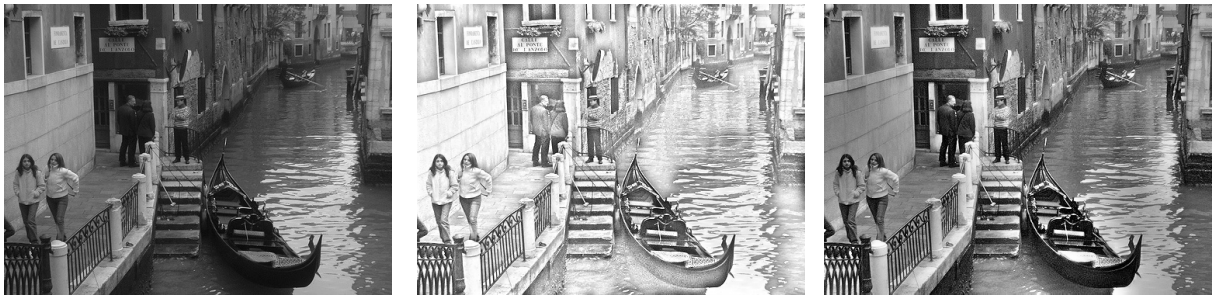
$$Image(I) = Illuminance\,(B) \times Reflectance\,(R). \tag{1}$$

Reflectance of objects in a scene is a quantification of what luminous energy the surface of an object can reflect back regardless of the sources of this energy. So reflectance depends on both geometric and textural properties of surfaces of various objects in a scene. Image edges, in particular, are associated with areas of high curvature of the surfaces of objects in an image as seen from a camera. Illuminance, represents the amount of light that reaches all objects in an image reduced by scattering on the surfaces in the image scene. Hence, illuminance, or simply luminance, does not necessarily capture correctly the effects of light scattering when the space between the scene and the camera contains anything other than clean air or, ideally, vacuum. Consequently, methods aiming to solve (1) are ideally suited for visibility improvement in turbid, or foggy environments.

Unlighting methods aim at solving this under-determined problem by approximating B and R from I. A good classical and with a lengthy history example is *Retinex* Land (1964), Land et al. (1977) modeled after the color perception system human vision. We use a modern and relatively fast variation of it, the **Variational Retinex (VR)** (Fu et al. (2016)). The other method we use is Illumination Normalization (Upadhyay et al. (2018)).

After taking logarithms of both sides of (1), the right-hand side has two terms which are approximated by means of a constraint optimization algorithm. This is the key idea of VR. The assumption for the solution or the constraints more precisely, are that solution should be such that regions with small value changes in R occupy most of the image, and what remains is edges and that the overall difference $I - R$ must be small. The optimization uses two weights or free parameters which control how much each of these constraints weigh in the final outcome. Clearly, the accuracy of the solution is heavily influenced by the choice of these two parameters. Prior knowledge of imaging conditions can possibly help to better choose the values of these two parameters. In our experiments, we used the MATLAB implementation provided by the authors of Fu et al. (2016) to acquire the reflectance estimation of the scenes (Figure 1 (b) and (d)). A major challenge in this implementations is finding optimal values for the two free parameters which control the minimization process.

On the other hand, the **Illumination Normalization (IN)** operator is a deterministic method, so it is faster than VR, thus, suitable for real-time applications of visibility improvement (Upadhyay et al. (2018)). The IN operator uses a multiscale transform to brake down an image into a individual components which, if put together, reconstruct the original. The IN operator takes each individual component and normalizes it in a way that the effect of illuminance at each scale is canceled out (Figure 1 (c) and (f)). The main effect of the transform is that the lower-scale components of this transform become more flat in terms of brightness, whereas the higher scale components which primarily carry the details of texture and edge information become more prominent. As a result, details necessary to object detection become more visible as if the image was acquired with adequate and uniform illumination. In Upadhyay & Papadakis (2019) the last two authors give a set of mathematical arguments proving that the outputs of IN operator contain information on shapes which is essentially the same with the information carried by a well-illuminated image of the same scene.



(a) Raw (original) (b) VR reflectance (c) IN enhanced

(d) Raw (original) (e) VR reflectance (f) IN enhanced

Figure 1: VR and IN image enhancement methods generate images where details, necessary to object detection, can become more visible. Given any raw input (**1st column**), we use the VR method to acquire an estimate of the scene's reflectance (**2nd column**) and the IN method to obtain a derivative image of the scene where the effect of illuminance has been reduced significantly (**3rd column**). In section 2.5 we state our premise that such image derivatives can be perceived as *virtual imaging modalities*, as the informative content they create may contain sufficient complementary information, with respect to the original image. In section 3 we demonstrate how state-of-the-art machine learning algorithms can be trained in tandem on the raw and artificial imaging modalities to learn representations that effectively harness the complementarity of information of virtual modalities (Original images are from the VOC2007 dataset).

2.2 Object detection

In an automatic **object detection task**, provided an input image and a pre-determined set of **target object classes** (e.g., person, car, dog etc), a system is expected to designate the smallest possible rectangular image regions containing an object from the target classes (**detection regions**) in a tight manner and also to predict the correct class label for each detection region. The fact that the number of possible detections per image, as well as the range of classes where those detections fall into, is not fixed, adds an extra degree of difficulty to this, supervised learning[1], task. Indeed, it is common to observe that a detection region contains objects from more than one of the target classes. In such a case, an efficient system should be able to assign all the appropriate labels to this same detection region. In this and the following sections we discuss the use of artificial modalities as inputs in DNN-based object detectors.

With a variety of applications, ranging from security (e.g., face detection and action detection) to industrial quality control (e.g., machine inspection, aerial surveillance and image analysis) and even commercial (e.g., visual search engines, autonomous navigation systems, point and shoot cameras), object detection holds an exceptionally dominant place in the efforts of computer vision researchers' community.

2.3 Modern object detection and classification

Our discussion, from now on, presents design ideas for DNNs which are fairly generic, despite the use of object detection as a case study. The ideas we present can easily be adopted for any multiclass, or one-class classification NNs with images as input data.

Notable early approaches to solve the object detection include the fast Viola-Jones framework Paul Viola (2001) and the combination of Histogram of Oriented Gradients (HOG) with Support Vector Machines (SVM) Dalal & Triggs (2005). During the recent years, artificial Neural Networks as **Deep Neural networks (DNNs)**, and especially their sub-class of **Convolutional Neural Networks (CNNs)**, have proven to be very successful on supervised learning image analysis tasks, including object detection. One of the first DNN approaches proposed the use of CNNs with multi-scale sliding windows scanning the input image Sermanet et al. (2014) which evolved into the various versions of YOLO (Redmon et al. (2015, 2016), Redmon & Farhadi (2018)) and RetinaNet(Lin et al. (2017)). These approaches are classified as **one-stage** methods for object detection. An alternative approach proposed the use of two stages, where regions with high probability to contain objects are singled out first, in what is known as a **region-proposal stage**. Then, a classifier examines each of these regions, to come up with a final decision on whether an object appears in the region and from which target class Girshick et al. (2014), Girshick (2015), Ren et al. (2015). There is a trade-off between processing time versus detection performance in both these approaches and utilizing the whole spectrum of their potential remains a very active domain of research.

2.4 Faster R-CNN

In the original settings of Region-proposal object detection, given an image input and a set of target object classes which the system is trained to recognize, a region-proposal detection system initially generates a feature representation of the input. Then, a binary "objectness" classifier examines this representation, to select **Regions of Interest (RoIs)** on the image, i.e. rectangular image regions where high confidence arises that one object from, at least, one of the target classes, is present. Subsequently, a learned representation for each RoI propagates, independently, into a second classifier that yields two final decisions. Primarily, for each ROI, this classifier gives 'confidence score' per target class, corresponding to the system's belief that an object from the specific target class appears in this ROI. Usually, one more confidence score is estimated, corresponding to class background, i.e. confidence that the ROI does not contain any object. Secondarily, for those target classes that the ROI receives relatively high confidence score to contain a corresponding object, the classifier attempts to adjust the dimensions and location of the rectangular ROI, aiming at a final **detection region** that captures the object in a tight-fit manner.

This, region-based, approach was introduced in Girshick et al. (2014) as **Region proposal CNN (R-CNN)** and quickly evolved into the Fast Girshick (2015) and finally the **Faster R-CNN** Ren et al. (2015). The main innovation of the latter, as compared to its region-based predecessors, is that the system addresses both region proposal

[1] In machine learning terminology, **supervised learning** refers to the process of training an algorithm by providing a 'labeled' training data-set. This means that, each data-set instance during the training is a pair of an input (typically an array of descriptors) and the 'correct output' that the algorithm is expected to return (**supervisory signal** or **label**).

and classification challenges resorting exclusively to CNNs, assisted by ranking operations and thresholding rules w.r.t. estimated 'confidence' values. In short, Faster R-CNN has two main components (Figure 2). The input image

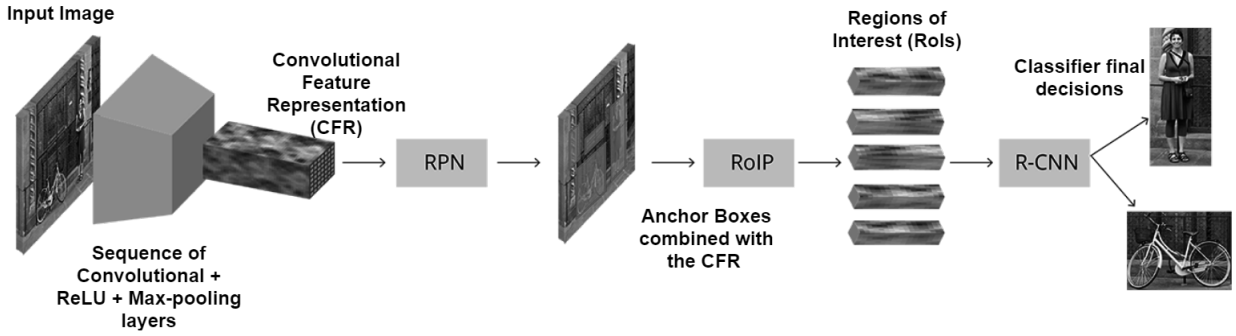


Figure 2: The Faster R-CNN object detection model depends exclusively on convolutional layers to tackle both Regions of Interest (RoIs) detection and Region regression-classification. The input image assumes its Convolutional Feature Representation (CFR), propagating through a sequence of convolutional layers with ReLU activation and periodical max-pooling operations, which reduce the spatial dimensions. The Region Proposal Network (RPN) receives the CFR and again through convolutional layers singles out Regions of Interest (RoIs), i.e. rectangular regions of the image with high 'probability' of an object to exist. Finally, the RoIs get re-sized and vectorized to fit as input to the R-CNN classifier, where through Fully Connected (FC) layers the final classification decision is returned.

propagates through a sequence of (trainable) convolutional layers which transform it into a **(convolutional) feature volume**[2] (3-dimensional array of descriptors), preserving the spatial organization, with periodical sub-sampling operations reducing its overall spatial dimensions. A (convolutional) **Region Proposal Network (RPN)**, utilizes this deep feature extraction to identify rectangular sub-areas of the image that 'probably' contain one or more objects from the target classes (RoIs), completing the first component. For the second component, the feature volume's portion that corresponds to each RoI is extracted (RoI Pooling (RoIP)) and propagates through a regression predictive model, implemented using fully connected layers (R-CNN), that assign the final target class confidence scores (including a 'rejection confidence score') and the final corrections of the RoIs location and dimensions.

One of the key novelties for the generation of RoIs in Faster R-CNN is the introduction of a deterministically constructed set of reference bounding boxes, called **anchor boxes** (Ren et al. (2015)). These boxes have fixed, predetermined, sizes and are (practically) uniformly positioned on the input image. There are many of them and overlap with one another. Any region candidate for becoming a ROI during the RPN stage, is referenced back to a unique anchor box, with a specific center location and dimensions with respect to the original input image I. If the dimensions of a convolutional feature volume, generated from a specific input are $W(width) \times H(height) \times D(depth)$ in pixels, for each anchor location $(i, j) \in [1, 2, ...H] \times [1, 2, ..., W]$ the system uses a set of rectangular anchor boxes, $S_{anchors}(i, j)$, with a variety of predetermined dimensions. In our implementation, we use nine anchor boxes per pixel (i, j), one square and two equal rectangles, landscape and portrait, each one of them with three different area sizes.

The Faster R-CNN architecture, and more specifically the VGG-16 based which is one of its most representative implementations, constitutes the basis for our *virtual multimodal* architecture. In the next sections we will review in detail the mechanisms of Faster R-CNN's components and will discuss the changes we propose, aiming in utilizing the artificial multimodal inputs.

2.5 Multimodal Faster R-CNN with artificial modalities

The success of DNNs and CNNs on supervised learning tasks using single modalities, motivated efforts to further improve the performance of such systems by combining multiple modality inputs (**multimodal learning**). (Ngiam et al. (2011), Srivastava & Salakhutdinov (2012), Eitel et al. (2015)). A common element in all these, recent, deep

[2]Often referred as feature representation in the literature.

multimodal approaches is that multiple modalities need to be a-priori available with the data-set. However, there are many situations where only one modality (e.g., monochromatic images) is available. In this section, and our experimental results in section 3, we demonstrate how image enhancement methods, such as VR and IN, derive scene perspectives (see Figure 1) that can complement the, often, limited information in the original (raw) modality. We treat the IN and VR derivatives as virtual imaging modalities, derived by the single available raw image modality.

We introduce a deep feature fusion architecture, the **virtual multimodal Faster R-CNN**, that combines the complementary information found in the original images and the derived modalities, improving detection and classification accuracy. Our proposed architecture is designed and implemented as a region proposal network, based on the popular Faster R-CNN model. However, its main innovation, fusion at the feature representation level using **derived artificial imaging modalities** when only one data modality is available, can be easily extended to multi-scale window networks. This remains part of our ongoing research.

2.6 Generating complementary information from virtual imaging modalities

The algorithmic design we propose in this section, relies on one premise:

> Image derivatives, generated from raw images using powerful image enhancement methods such as VR and IN, can be perceived as **virtual imaging modalities**. The informative content they create may contain sufficient complementary information, with respect to the original image. State-of-the-art machine learning algorithms can be trained in tandem on the raw and artificial imaging modalities to learn representations that effectively harness the complementarity of information of virtual modalities. Moreover, when the conditions during image acquisition had been adverse, e.g., under low/uneven illumination, the artificial modalities can compensate for the loss of information.

Of course, this is also true for the case of non virtual modalities. For example, Lidar and Sonar can reveal the shape of objects even in zero visibility conditions, where image acquisition in the visual spectrum is useless. Successful methods have been developed to combine complementary information from different acquisition sensors, as we discuss in the following sections. In this work we focus on the following problem, which is of pivotal importance in the common case where only one imaging (original) modality is available:

> Can the complimentarity of information from original and virtual modalities be leveraged to improve detection and recognition accuracy of state-of-the-art algorithms?

2.7 Virtual multimodal Faster R-CNN

We adopt the architecture of the very widely used, VGG-16 based, Faster R-CNN object detector. We independently train clones of this model and then transfer trained sections into a fusing architecture, which we design explicitly for accommodating the complementary informative input from IN and VR. We should note, though, that that this strategy can easily be extended to different Faster R-CNN set-ups, and even to other popular, not region-based, object detection architectures (e.g., YOLO (Redmon & Farhadi (2018)) and RetinaNet(Lin et al. (2017))). For simplicity, we describe in detail only the fusing architecture of our **Bi-modal Faster R-CNN** set-up. However, the generalization to accommodating a larger number of, physical or virtual, modalities, such as the **Tri-modal** version that we use in our experiments (section 3), is fairly straight forward.

Starting with the original (**raw**) data-set, which consists of monochromatic (e.g., gray-scale) images (**RD**), we select and apply the image enhancement method of our choice. The derivative data-set (**SD**) will serve as additional, virtual imaging modality. Using the real RD and the derivative SD imaging modalities, we train two independent, but identically structured, **Uni-modal VGG-16 Faster R-CNN** networks, as described in the following paragraph.

Uni-modal VGG16 Faster R-CNN

For each training session (RD and SD), the corresponding image becomes the input of a **Convolutional Feature Representation (CFR)**. This mapping is implemented by a series of 13 convolutional layers, each one followed by Rectified Linear Units (**ReLU**), with the max pooling operation activated after selected layers, as shown in Figure 3. The number of filters increases after every few layers, as is common in CNN architectures. This process transforms

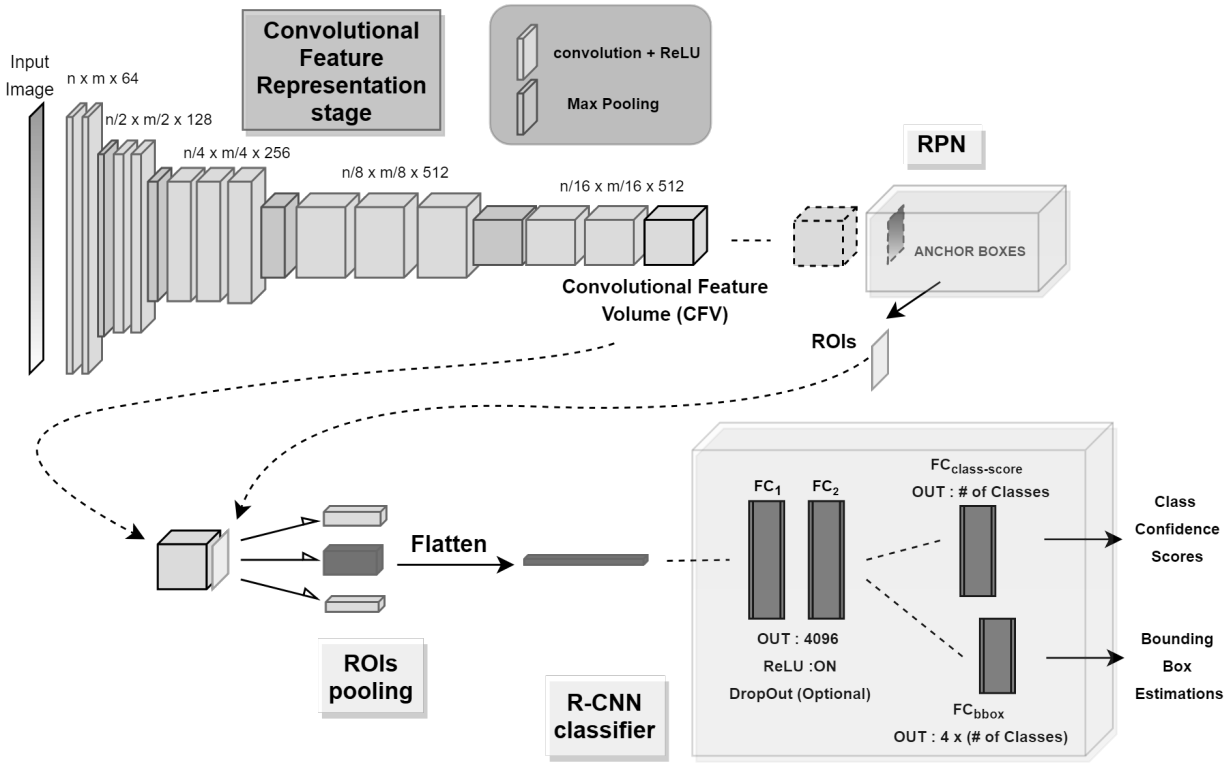


Figure 3: The Unimodal VGG16 Faster R-CNN propagates the image input through a sequence of convolution layers (Convolutional Feature Representation (CFR) stage), periodically reducing the spatial dimensions with Max Pooling, generating the (convolutional) feature volume. This representation enters the RPN which, using the pre-determined anchor boxes as reference regions, selects ROIs, i.e. regions of high 'objectness'. Finally, each ROI is vectorized and pushed into the R-CNN classifier stage where, through a succession of Fully Connected (FC) layers, the system arrives at the final decision: object from which classes appears in the detected region (if any) and final location and dimensions adjustments to fit the object tightly (if any).

the original input into a 3-dimensional array, the (convolutional) feature volume[3]. The max pooling operations (red color layers in Figure 3) divide the CFR component in five sections. The spatial dimensions of the feature map are bisected as the volume of processed data passes forward, from one section to the next. We label each convolutional layer as *convM.P*, where $M = \{1,2,3,4,5\}$ corresponds to the section where the layer belongs and P is its position in the specific section. For example, *conv*1.1 refers to the very first convolutional layer that receives the input image, while *conv*5.3 refers to the last convolutional layer in the CFR component. The coefficients throughout the CFR component, both for the convolutional filters and the biases, are all initialized using the corresponding coefficients of the VGG-16 network pre-trained on the (RGB) Imagenet data-set (Deng et al. (2009)) (see following paragraph for more details on the model parameters' initialization). The spatial dimensions of all convolutional filters (**kernels**) used in the CFR component are the same, 3×3. We allow the training to update all layers' filter coefficients. Furthermore, all convolution operations act on the previous layer's output-array with zero-padding applied as needed, so that the output's spatial dimensions do not drop due to the convolution operation Thus, for an image input of size $n \times m$, the output of the CFR stage, i.e. the output of the *conv*5.3 layer, is a convolutional feature volume with dimensions $(m/2^4) \times (n/2^4) \times 512$.

The convolutional feature volume enters the next stage of the unimodal network, the RPN. In this stage, one more convolutional layer ($RPN_{3\times3conv}$), composed by 512 filters of size 3×3 and followed by ReLU operation, acts on

[3]Often called **feature map** or **feature representation**

the feature representation. Once more, appropriate zero padding is used to retain the map's spatial dimensions. The $RPN_{3\times3conv}$ output enters two independent 1×1 convolutional layers, each with enough channels to output objectness scores (RPN_{class}) and bounding box predictions (RPN_{bbox}) respectively, for all corresponding anchor boxes of the image. More precisely, since we use $|S_{anchors}| = 9$ anchor configurations, the RPN_{class} layer will have $2\cdot9 = 18$ output channels (a pair of confidence scores per anchor representing confidence of being an object or being background) while RPN_{bbox} will have $4\cdot9 = 36$ output channels, corresponding to 4 regression bounding box (**bb**) adjustments for each anchor ($\Delta_{x_{bb-center}}$, $\Delta_{y_{bb-center}}$, $\Delta_{bb-width}$ and $\Delta_{bb-height}$ respectively).

An $(m/2^4)\times(n/2^4)\times512$ feature volume yields a total of $M = (m/2^4)\times(n/2^4)\times9$ candidate ROIs. Each of them has now been assigned objectness scores and bounding box adjustment values. However many of these regions are in practice 'significantly' overlapping due to the anchors' construction. At this point, a process known as **Non-Maximum Suppression (NMS)** takes place. The aim of this process is to eliminate most of the overlapping ROI candidates, allowing those with the highest objectness scores to prevail. The list of candidate regions is re-arranged in decreasing order with respect to objectness score[4] and a predetermined NMS threshold, $NMS_{thres} \in (0,1)$, comes into play. The regions are compared on the basis of their **Intersection over Union (IoU)** ratio; the IoU of two image regions A and B is defined as the ratio of their intersection area over the area of their union $\left(\frac{|A\cap B|}{|A\cup B|}\right)$, measured in pixel units. Each RoI candidate region that demonstrates IoU$> NMS_{thres}$, with a region found higher in the list, gets discarded. Through this process, the RPN yields the final list of ROIs.

ROIs, through their reference anchor and the quadruple of region adjustments, should be regarded as sub-regions of the original, $n\times m$, input image. However, ROIs can also be projected onto the convolutional feature volume, defining feature volume '3D-slices'. These 3D-slices are extracted and interpolation in space is applied, to achieve a uniform size of $7\times7\times512$, completing what is commonly referred to as the **ROI pooling** step. These, new equally sized, **RoI feature volumes** assume now the role of RoI descriptors for the final, R-CNN classification, stage.

Each ROI feature volume that comes out of the ROI pooling process is vectorized and pushed into the Fast-RCNN classifier. Here, two consecutive, **Fully Connected (FC)**+ReLU layers, FC_1 and FC_2 respectively, transform the vector representation, both producing outputs of size 4096. The (random) **drop-out** operation, a very common sparsification (regularization) technique that allows only an arbitrary percentage of the coefficients to be fine-tuned at each training iteration (to reduce over-fitting), may be used here[5]. Note that certain sparcification (a reduction in the number of small in magnitude thought as insignificant values) already has taken place, through the ReLU and the max-pooling operations. This additional sparsification step can compensate for some residual over-redundancy of the feature mapping mechanism, built before the RPN (original sparsification comes from max-pooling operations and the ReLU non-linearity). We note here that, the drop-out operation is commonly regarded as regularization of the back-propagation process. We believe that both viewpoints offer a different, useful perspective.

At last, the feature vector output of FC_2 enters two parallel, independent FC layers, that yield the two final system predictions. The first ($FC_{class-score}$) generates the softmax[6] confidence scores per class, including one background class. The second (FC_{bbox}) yields the final regression bounding box adjustments, intended to make each detection region as tight as possible around the predicted object. Thus, in a detection task where the number of target classes is C, the final output of any original input is a summary 'report' for a number of RoIs, say N, presented in two tables:

$P \in [0,1]^{N\times(C+1)}$: contains the $FC_{\text{class-score}}$ layer output (softmax class scores). Each row corresponds to a RoI. Each column corresponds to a target object class, plus one more column for class 'background'.

$B \in \mathbb{R}^{N\times4\cdot(C+1)}$: contains the FC_{bbox} layer output (box adjustments). Each row corresponds to a RoI. Every next quadruple of entries in a row corresponds to one of the target classes, again with the addition of an extra quadruple for 'background'.

[4]Note that only the confidence score of being an object is used

[5]In our experiments we observed that in this particular situation drop-out does not allow the model to achieve its best performance. Thus, we adopted an early-stopping-like technique, by allowing the training to continue for many epochs after convergence on the training set has been achieved, and then choosing the model that performs best on the validation set as representative of the scheme

[6]Given a vector $u = (u_1, u_2, ...u_K) \in \mathbb{R}^{\mathbb{K}}$ the softmax of u is composed by the elements $s(u)_i = \frac{e^{u_i}}{\sum_{k=1}^{K} e^{u_k}}$.

Initialization

During training all but the predefined, if any, filter entries the system will use will be learned via a process which forces them to change in a way that a certain input from the training set must be matched to a certain output. One classical approach is to give random values to those entries. Assigning values to any filters entries or weights to fully connected layers at step one of the training is called *initialization*. One of the main difficulties when training deep CNNs is that the convolutional layer coefficients, if randomly initialized, tend to over-fit the training data-set. Furthermore, for many computer vision tasks, even data-sets composed by hundreds of thousands of instances can be considered relatively small to represent adequately the input space in its generality. These factors can lead, not only to very long training sessions but, even worse, to models where the loss-function optimization process (see next paragraph) gets trapped inside a 'local minimum', or reaches a 'saddle point', quite far from a model that would generalize efficiently on un-seen data. A widely accepted solution to this problem, akin to the 'human experience', is **transfer-learning**. According to the transfer-learning strategy, new architectures that have to be trained on relatively small datasets, can benefit by 'borrowing' convolutional or FC layers' coefficients, previously trained on much larger data-sets. There is evidence that this transfer of coefficients works even when the tasks are not exactly the same. We use the, openly available, pre-trained VGG 16 coefficients from Simon et al. (2016), to initialize the filter coefficients in all convolutional layers in the CFR stage of our unimodal sessions. Ideally, in our enhanced data SD-unimodal session, we would rather use an initialization using coefficients obtained from pre-training VGG16 with an enhanced version of the ImageNet data. However, due to computational limitations arising from the large size of the ImageNet data-set, this has not been feasible during our experiments.

The coefficients for all the layers following the CFR stage, convolutional and fully connected, are randomly initialized, using normal distributions random inputs, but ignoring the values that are more than two standard deviations (**stddev**) away from the mean. Values outside this interval are discarded and re-drawn. The parameters of the normal distributions we use in our implementation are: *(i)* mean= 0 and stddev= 0.001 for the FC_{bbox} layer and *(ii)* mean= 0 and stddev= 0.01 for all remaining layers.

Loss function and optimization

Supervised machine learning apparatus is driven by what is known as the **loss function**, that measures the discrepancy between the system's predictions and the correct 'labels' (**ground truth** values). Loss function plays the role of the objective function in an optimization scheme and is evaluated at each training iteration. To control the quality of the approximation of the model's predictive function (here the predictive function is a scoring function), the model aims at optimizing (minimizing) the loss function, by updating the model's trainable variables as needed (**backpropagation**). However, simply minimizing the loss function does not necessarily lead to a good predictive model. During the training phase, therefore also during the loss function minimization process, only instances from the training set are examined. But the evaluation of a model's predictive accuracy is done by measuring its predictive ability on the separate test set, after the model's training has been completed (usually on the basis of loss function convergence). The ability of a model to predict the labels of instances that never appear during its training phase is commonly known as the model's **generalization capability**. Good generalization capability is the key ingredient that needs to complement a successful loss function minimization scheme, in order to obtain an efficient predictive model. These issues need to be very carefully considered when designing the model architecture, the loss function as well as the optimization scheme.

For the Uni-modal Faster R-CNN, we use the loss function defined in Ren et al. (2015). In short, the loss for an image input is defined as the sum of four partial losses, evaluated in it's two prediction stages:

$$Loss_{Total} = Loss_{cls}^{RPN} + Loss_{reg}^{RPN} + Loss_{cls} + Loss_{reg}$$

The first two terms of the sum (**RPN loss**) refer to the classification (log loss) and bounding box regression loss of the RPN respectively. The last two terms (**classifier loss**) steer the final classification and detected region regression loss. The classification loss terms penalize the difference between the systems confidence that a region contains an object and the actual objectness label of the region (0 if the region does not contain object and 1 if it does). Both regression loss terms measure the 'mismatch' of an estimated detection box with the corresponding ground truth box, at some location of the image that contains an object.

A critical factor of the training process is the balanced mix of anchor boxes used to evaluate RPN loss. Anchor boxes can be divided into two categories. Those that do overlap 'significantly' with ground truth box annotation of an

object from the target classes, and those which do not. We call the former **foreground anchor boxes** and the latter **background anchor boxes**. Since objects from the target classes in most images occupy a small portion of the total image area, the RoI candidate anchor boxes dealt by the *RPN* that are background anchor boxes highly outnumber the foreground ones. This may bias the model towards the background 'class', if all anchor boxes are allowed to enter the RPN loss. To avoid this, during training, the set of anchor boxes is divided into two groups:

positive: Those that have an IoU greater than 0.5 with a ground-truth region of the image, and

negative anchors: Those that have $0 \leq$IoU< 0.1 with all ground-truth regions of the image.

Then, a balanced random sample of 256, both positive and negative, anchors is selected. In the evaluation of the RPN loss terms at each training iteration, only the mini-batch anchor boxes are used in the calculation of $Loss_{cls}^{RPN}$ and, from those, only the positive anchor boxes and their 'closest' ground truth regions are used in the calculation of $Loss_{reg}^{RPN}$.

Using the $Loss_{Total}$, Stochastic Gradient Descent (SGD) with momentum (Sutskever et al. (2013)) is used as the **optimization scheme**, with the momentum value set at 0.9. The initial value for the **learning step** is set to 0.001 and decreases in a step-decay manner, multiplied by a factor of 0.1 every $70,000$ training iterations.

Bi-modal VGG16 Faster R-CNN

We allow the two independent training sessions, for the RD and VR Unimodal Faster R-CNN respectively, to converge. Then, we transfer the CFR stage coefficients and the $RPN_{3\times3conv}$ coefficients of the best performing agent[7] from each, into the **Bi-modal Faster R-CNN** (Figure 4), which we design and train as follows:

The RD physical modality and the SD virtual modality are used as inputs to two CFR streams, arranged in parallel. Each stream is initialized with the coefficients learned and transferred from the corresponding RD and SD Unimodal sessions, from the previous step. The coefficients of these two, independent, streams will remain un-effected by the (optimization guided) updating process, throughout the whole bi-modal training session. The feature volumes generated by **RD-CFR** and **SD-CFR** enter the RPN stage, where two separate $RPN_{3\times3conv}$ layers receive them, the $RD-RPN_{3\times3conv}$ and $SD-RPN_{3\times3conv}$ respectively, both with the same set-up as in the unimodal case[8], but each initialized using the pre-trained $RPN_{3\times3conv}$ coefficients of its corresponding unimodal peer. The outputs of these two, parallel, convolutional layers, are concatenated (deep features fusion) towards the 'depth' dimension. The rest of the RPN stage maintains the same structure as in the Unimodal case, producing the RoIs of high objectness confidence, and the training remains the same. It is important to note though, that, since the depth dimension of the fused representation has doubled due to the fusion of the two feature volumes, the depth of the 1×1 filters in the two, parallel, convolutional layers that conclude the RPN stage, are also doubled. Now, the Bi-modal model has, hopefully, already used the dual information from the RD and SD fused representation volumes, created in the RPN stage, to better estimate the RoIs. For the R-CNN classifier stage, we perform a second deep feature fusion, by concatenating the, already generated, feature outputs from the RD-CFR and SD-CFR stages, towards the depth dimension. Then we (inversely) project the generated RoIs on this **fused convolutional feature volume (FCFV)**. We note that in this case, the representation 'slices' extracted in the RoIs pooling stage inherit the depth of the FCFR, which is twice the size of that in the uni-modal case. The rest of the Bi-modal architecture proceeds with the same manner as the Uni-modal network. The RoI feature representations are vectorized and enter the R-CNN classifier stage. However, since this time the vectorized RoI array has twice the size of the Uni-modal equivalents, the size of the embedding matrix in the FC_1 layer for the Bi-modal architecture needs to have dimensions $(7\times7\times2\times512)\times4096$. Once more, for FC_1 and FC_2 layers in the Bi-modal network, one can choose to use Drop-Out, probably to correct the redundancy of information, which may lead to over-fitting. However, in our implementation, in order to avoid the negative implications of fine tuning the model, we do not use drop-out but, once again, follow our Uni-modal 'early-stopping' approach. That is, we train until the model has safely converged and then we run validations using snapshots of the model, recorded at every $5,000$ training iterations, choosing the best-performing one as the optimal model. The final predictions of class confidence scores and bounding box estimations enter the same loss functions with those described in the uni-modal case.

[7] We save a 'snapshot' of the models' coefficients every $5,000$ training iterations. A validation is performed on the validation dataset using each coefficient's snapshot and the best performing is determined according to our 'over-all classes' evaluation metric, described in section 3

[8] 512 filters of size 3×3 followed by the ReLU operation

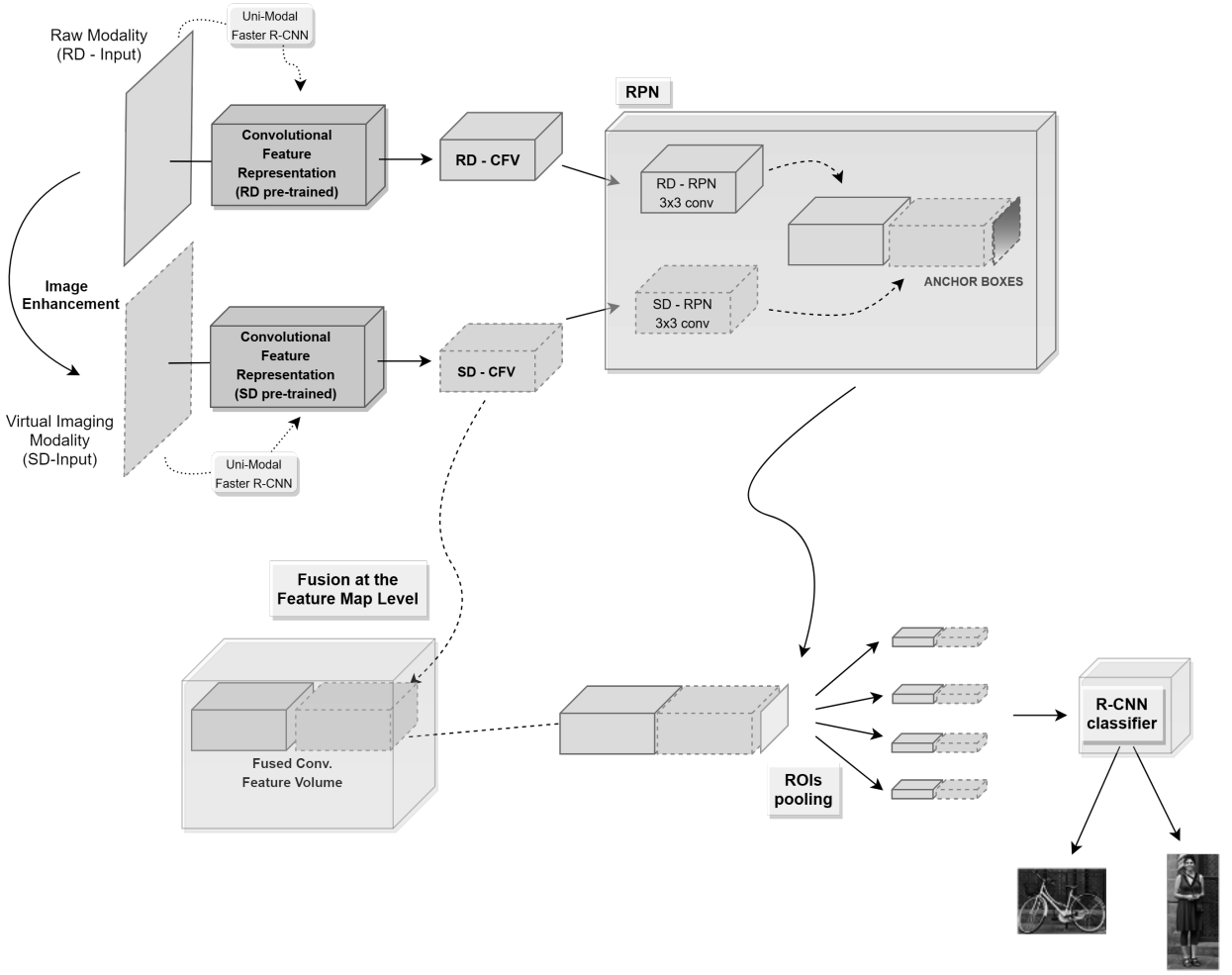


Figure 4: BiModal Faster R-CNN propagates the dual modality input through two parallel streams of identically structured, but independently pre-trained, CNNs. The CNN that receives the Raw (physical) modality (RD) is pre-trained on Raw images while the CNN that receives the virtual modality (SD) is pre-trained trained on processed images. The two streams produce two separate Feature volume representations. The two representations are concatenated (deep features fusion) and the new, deeper, volume enters the Region Proposal Network (RPN), allowing the system to estimate more accurate Region of Interest (RoI) detections. Finally, fused representations of these RoIs enter the classifier leading to more accurate label decisions.

Initialization of the Bi-modal Faster R-CNN

As we already mentioned, all the filter coefficients in the two parallel CFR stages, as well as the filter coefficients in the two parallel $RD-RPN_{3\times3conv}$ and $SD-RPN_{3\times3conv}$ layers, are initialized by transferring the corresponding coefficients of the best performing models from the Unimodal networks. Once more, we emphasize that these coefficients will remain fixed during the Bi-modal network training process, unaffected by the update which takes place after each training 'forward pass', in what is commonly known as the 'back-propagation' step in deep learning terminology. Apart from these, transferred, coefficients, all other coefficients, in all remaining layers of the Bi-modal network, convolutional and FC, are randomly initialized, using the same initialization set-up we used in the Unimodal training sessions. Furthermore, all these coefficients in the remaining layers, are allowed to get updated during each back-propagation step, until the model converges.

Before we close this paragraph, we need to mention one more thing. It would be possible to transfer the independently pre-trained coefficients of the FC_1 layers from the RD and SD Uni-modal networks, and combine them to initialize the FC_1 layer in the Bi-modal network. This could provide further generalizability gains to the Bi-modal network, and in extension to any multimodal network, especially since the size of FC_1 layer increases proportionally to the number of modalities used, but our postulate does not imply increase in the usable complementary information that is proportional to the number of used modalities. This type of modified multimodal Faster R-CNN initialization remains part of our ongoing experimentation.

3 EXPERIMENTS

We test our Multimodal Faster R-CNN implementation on two very popular, publicly available datasets suited for object detection, the **Pascal VOC 2007** (Everingham et al. (n.d.*a*)) and **VOC 2012** (Everingham et al. (n.d.*b*)) challenge databases. More specifically we follow the object **detection task** rules: Predict the bounding box and label of each object from the twenty target classes in an image unknown to the system being tested. As the challenges' creators state: *The main goal in these challenges is to recognize objects from a number of visual object classes in realistic scenes (i.e. not pre-segmented objects).* This can be treated as a supervised learning problem and a training set of labeled images, from twenty different object classes, is provided. The twenty object classes that have been selected for both challenges are:

Person: person

Animal: bird, cat, cow, dog, horse, sheep

Vehicle: aeroplane, bicycle, boat, bus, car, motorbike, train

Indoor: bottle, chair, dining table, potted plant, sofa, tv/monitor

We run two independent experiments, comparing the accuracy of the Unimodal Faster R-CNN with raw image input (benchmark) against our Bimodal and Trimodal Faster R-CNN architectures, that use the IN/VR artificial imaging modalities. We follow the standard experimental set-up for the VOC2007 and VOC2012 detection tasks:

VOC2007 detection: We train on both the *train* and *validation* VOC2007 data-sets, composed by $5,011$ images containing $12,608$ object instances in total. We validate the model on the VOC2007 *test* data-set, composed by $4,952$ images containing $12,032$ object instances in total.

VOC2012 detection: Due to lack of available annotation for the VOC2012 *test* set, here we train only on the VOC2012 *train* data-set, composed by $5,717$ images containing $13,609$ object instances in total. Then, we validate the model on the VOC2012 *validation* data-set, composed by $5,823$ images containing $13,841$ object instances in total.

3.1 Precision over recall evaluation per object class

In all the Faster R-CNN based network's we use in our tests, a usually large number of RoIs are selected from the RPN classification as significant object-containing RoIs. Each one of these RoIs enter the R-CNN classifier, which predicts the final detection region and gives to it a confidence score per class. It is crucial to understand that this NN is a scoring system and the score is its belief that a certain region selected among those that have been qualified by RPN contains an object from a certain class. In Figs. 7,8, and 9 the green detection boxes are such regions but those displayed are a minuscule fraction of all anchor boxes qualified by both RPN and classifier. Recall, that RPN assigns to each anchor box a similar "objectness score"

Given some **confidence-threshold** value ct_A for class A, a detection region, say BB_{detect} (recorded in the array B), is declared as a **positive detection** of object from class A, if the class A confidence estimated by the R-CNN classifier, say $score_A^{BB_{detect}}$ (as recorded in the array P), exceeds the value of ct_A. If $10^{-5} < score_A^{BB_{detect}} < ct_A$ then BB_{detect} is

registered[9] as **negative detection**, i.e. the network's classifier decides that no object from class A is found in BB_{detect}. This rule, yields a classifier for each class and confidence-threshold value.

Now, for evaluation purposes, a positive detection region $BB_{pos-detect}$, for some ct_A, is considered a **True Positive (TP) detection** if it satisfies both following rules:

(i) $BB_{pos-detect}$ has an IoU> 0.5 overlap with a class A ground truth box, say GT_A

(ii) GT_A ground truth region is not overlapped (under the same IoU criterion) by another positive detection box of higher class A confidence

If at least one of (i) or (ii) is not satisfied, $BB_{pos-detect}$ is registered as **False Positive (FP) detection**. It is important to observe here that, the system needs to be 'chary' with its positive decisions. Indeed, in case of multiple 'valid' detections of the same object in an image, all detection regions, except from the most confident one, will be declared as false positives for the evaluation purposes[10] .

To obtain a comparable measure for the accuracy per class of all Faster R-CNN based networks we use in our tests, we adopt again the **Precision (Prec)** over **Recall (Rec)** standard metric. In probabilistic terms, for any class c and some confidence-threshold ct_c, we define

$$\begin{aligned} Prec_c = & P(\textit{detected region indeed contains object from class } c \mid \\ & \textit{system declared that the detected region contains object from class } c) \end{aligned}$$

which is also known as *positive predictive power* in biomedical tests. On the other hand, recall is the *sensitivity* of a system, i.e.

$$\begin{aligned} Rec_c = & P(\textit{system declared a detection region containing object from class } c \mid \\ & \textit{an object from class } c \textit{ indeed exists in this region}) \end{aligned}$$

where $P(A|B)$ represents the conditional probability of event A given B. For the purpose of extracting empirical probabilities of performance, precision and recall are given by the following formulas:

$$Prec_c = \frac{\text{Number of class c TP detections}}{\text{Total number of class c positive predictions}}$$

and

$$Rec_c = \frac{\text{Number of class c TP detections}}{\text{Total number of class c ground truth regions in the (test)set}}$$

Typically, lower precision is expected at higher levels of recall and vice versa. However, in many applications, it is more important to know the precision of a system even at low recall levels, or the recall at low precision levels. Furthermore, it can be of great importance to know what is the best precision one can expect when recall below a specific level cannot be tolerated, or the best recall one can expect when precision below some level is unacceptable. To help understand those concepts we stress that recall is sensitivity and precision is the positive predictive value of the system for screening tests in medicine.

We plot corresponding curves for all significant confidence-threshold values. This curve represents the, per class, accuracy of the multi-class object detection model. Depending on the use of each modality, physical or virtual, we obtain the following networks, of which we compare their individual performances:

Raw: The Unimodal Faster R-CNN network, described in section 2.5, using only the raw grayscale version of the data both for training and testing.

Bimodal.Raw.IN: The Bimodal Faster R-CNN architecture, described in section 2.5, fusing the raw grayscale and the virtual modality induced by IN version of the data, both for training and testing.

[9]It is common for this minimum score threshold value to be set higher than our 10^{-5} choice, so that less regions will enter the evaluation stage as negative, leading to faster evaluation calculations. However, we purposefully decide to set this threshold in such a low value that (almost) all RoIs will enter the evaluation stage, providing us with a more complete picture of the over-all precision/recall trade-off of each tested model.

[10]In statistics language this means that all rank 2 and higher detection regions that effectively capture the specific object will enter the evaluation as false positives

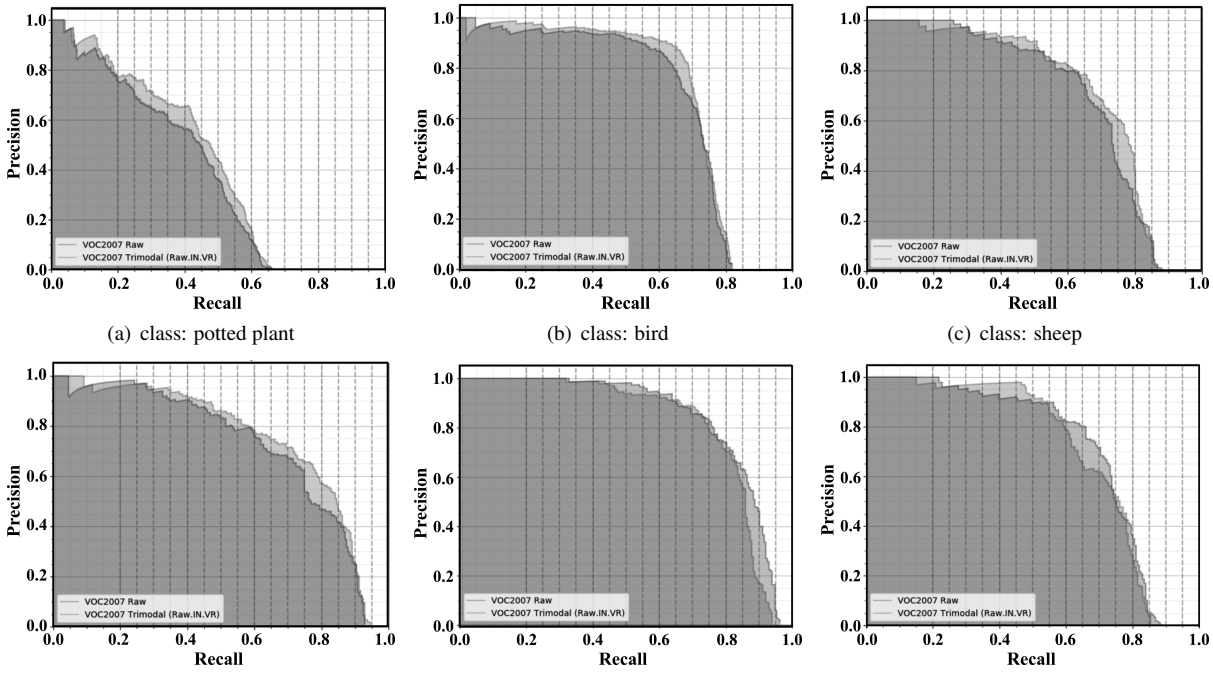
Bimodal.Raw.VR: The Bimodal Faster R-CNN architecture, fusing the raw grayscale and the virtual modality induced by VR version of the data, both for training and testing.

Trimodal.Raw.IN.VR: The Trimodal Faster R-CNN architecture, fusing the physical raw grayscale and both the IN and VR virtual modalities.

Results per class

In Figure 5 we present the precision over recall curves (for a few classes) from the Raw Unimodal Faster R-CNN detector against our Trimodal Faster R-CNN architecture. In most cases, class accuracy improves anywhere from somewhat to significantly, when we use the proposed Multimodal networks (Bimodal or Trimodal Faster R-CNN) relative to the raw-Unimodal model (e.g., for classes *potted-plant*, *bird*, *sheep* and *sofa* in Figure 5(a)-(d)). The gain in precision with the Trimodal architecture is anywhere between 0.02% up to 18.16% in most recall levels. On the other hand, there are object classes, e.g., the class *bus* (Figure 5(e)) where the Trimodal network results in a lower precision than its raw-Unimodal peer. We also find object classes, such as *dining-table* (Figure 5(f)), where performance is mixed

We observe a similar behavior with the VOC2012 database. Thus, in order to reach a conclusion on whether an overall accuracy gain arises from our Multimodal architecture in both these tasks, relatively to the Raw-Unimodal model, we propose to use a descriptive augmented metric, to represent performance of each model over all object classes.

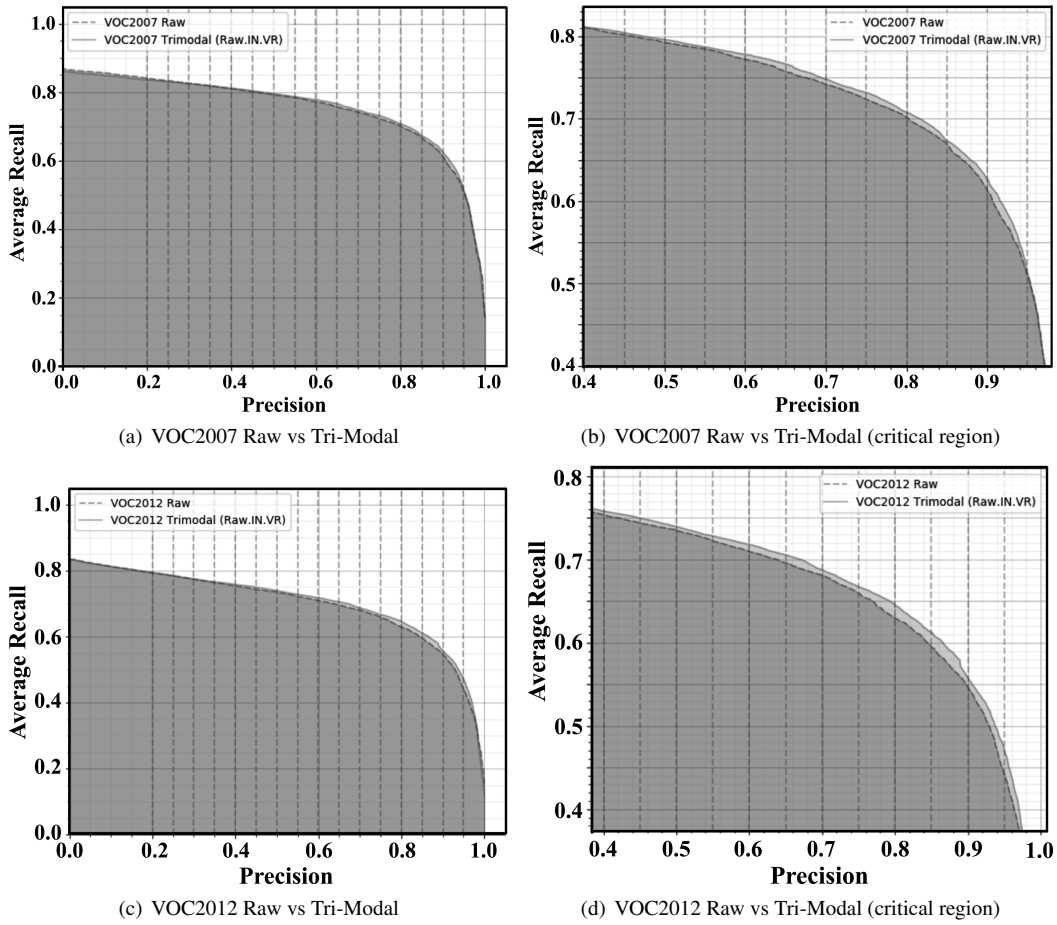


(a) class: potted plant (b) class: bird (c) class: sheep

(d) class: sofa (e) class: bus (f) class: dining table

Figure 5: Precision over recall, from VOC2007 object detection task, for classes *potted plant* (a), *bird* (b), *sheep* (c), *sofa* (d), *bus* (e) and *dining table* (f). Both models, the Raw Unimodal Faster R-CNN (blue) and the Raw.IN.VR Trimodal Faster R-CNN (orange), have been trained on the corresponding version of the VOC2007 *train+validation* dataset. Their performance is evaluated on the corresponding VOC2007 *test* data-set. In most object classes, e.g., (a)-(d) here, we observe a significant increase of precision when we use our Raw.IN.VR-Trimodal model. However, there are cases, e.g. (e) here, where the virtual modalities based Trimodal architecture demonstrates lower precision, and others where performance is mixed (f).

Each model's performance can be studied on a three-dimensional space, with Recall (**r**), Precision (**p**) and Object Class (**c**) defining the three dimensions. The range for the first two dimensions, r and p, is the interval $[0, 1] \subset \mathbb{R}$.



(a) VOC2007 Raw vs Tri-Modal

(b) VOC2007 Raw vs Tri-Modal (critical region)

(c) VOC2012 Raw vs Tri-Modal

(d) VOC2012 Raw vs Tri-Modal (critical region)

Figure 6: All-class average recall *(AR)* plotted as a function of precision *(p)* for the VOC2007 (**top row**) and VOC2012 (**bottom row**) multi-class object detection tasks. The blue curve is drawn for the Raw-Unimodal Faster R-CNN benchmark. The orange curve is drawn for the proposed Trimodal Faster R-CNN architecture, where we combine deep features from the Raw and both virtual IN and VR modalities. Our Raw.IN.VR-Trimodal detector achieves significant increase in the (expected) recall on the largest part of the precision range. In the VOC2007 case, apart from a low region below 37% precision, the Raw.IN.VR-Trimodal detector achieves an increase of the expected (overall object classes) recall from 0.09% to 1.42% (at 90% precision). In the VOC2012 case, the proposed Raw.IN.VR-Trimodal architecture achieves significant increase in the (expected) recall on the whole precision range, from 0.15% up to around 1.76% (at the neighborhood of 85% precision), relatively to the Raw-Unimodal benchmark.

Dimension c is discrete and assumes any value from the set $\{1,2,...,C\}$, where C is the number of target object classes. In our VOC2007 and VOC2012 tasks, $C = 20$. For any model for which the precision recall values (curves) have been evaluated for all object classes, we define the function $r(c,p)$ as

$$r(c,p) = \text{recall value at precision level p for class c}$$

Note here that, precision does not necessarily cover all the available $[0,1]$ range, however there exists a meaningful correspondence between all achieved precision ranges between the different classes. Now, we define the **Average Recall (AR)** as

$$AR(p) = \sum_{c=1}^{20} emp(c)r(c,p) \tag{2}$$

where $emp(c)$ equals the relative frequency of appearance of object class c in each task's data-set, i.e. the ratio of class

c ground truth regions in the data over the total number of ground truth annotations in the data. Thus *AR* is the *average empirical expectation of recall r as a function of precision p*.

We use the *AR* function to evaluate the (expected) over-all performance of the Faster R-CNN based models. We plot the *AR* over precision curves from different models on the same graph, to compare their performance in terms of expected over-all recall at any level of precision. In both cases, on the VOC2007 and on the VOC2012 databases, for the task of multi-class object detection, we have observed a significant improvement in the AR, the class-average recall, for the largest part of the precision range, when we engage the virtual modalities with any of our Multimodal architectures, relative to the Raw-Unimodal benchmark model. However, the boost in expected recall is higher when we use our Raw.IN.VR-Trimodal Faster R-CNN architecture (Figure 6). More precisely, in the case of the the VOC2007 database, apart from a low region, below 37% precision, using our Raw.IN.VR-Trimodal detector we achieve an increase of the all class-average recall from 0.09% up to 1.42% (at 90% precision). In the case of the VOC2012 database, the respective increases are observed over the whole range of precision, and range from 0.15% up to 1.76% (at 85% precision).

4 CONCLUSIONS

Multiple physical modalities provide complementary information for artificial intelligence tasks. They work in such a way that when the information flow of one modality does not contribute sufficiently to a decision making process, additional modalities with play a complementary role which, will hopefully decrease disambiguation of this task. If only one physical modality can generate inputs, then there is only one way to proceed; maximize the extraction of information from the single physical modality inputs. In this work we propose that one specific pathway to achieve this goal is by creating virtual modalities, which generate artificial inputs from the single available input. The single physical input and the artificially generated derivative inputs are jointly fed to a Neural Network classifier. We do this in two ways, either by expanding the feature space or by utilizing the structure of the Neural Network classifier, which now receives the single output along with the virtual inputs and generates an expanded feature vector for any non-virtual input.

Within this framework, we have been able to provide preliminary experimental evidence to support the claim that the IN and VR operators increase the ability of the deep object detection NN to assign less ambiguous confidence scores, without imposing significant changes in the network architecture other than those necessary to accommodate the additional artificial modalities. The reader can see the result of this disambiguation in Figures 7,8 and 9.

Our results are preliminary since we have not ascertained the statistical significance of our findings, mainly, due to our intention to adhere as much as possible to the evaluation protocols of the Pascal VOC object detection competitions. The computational overload required for this task was another contributing factor for this decision. Moreover, the precision-recall curves of the R-CNN classifier performance show mild improvement with the fusion of the features of the virtual modalities we use. Of course, one may wonder whether the same or better improvement can be achieved with changes in the networks architecture. Although, this is quite possible, the improvement of discriminability seems to suggest that the artificial modalities inputs may even yield additional improvement. However, the huge variety in network configurations and design can lead to huge volumes of experiments that one cannot possibly perform in order to establish, beyond any doubt, the usefulness of the artificial modalities along with a physical one. Moreover, the identification of the correct parameters for the artificial modalities is another issue we did not touch upon in a systematic way (sensitivity analysis). In fact, in Fig. 9 the reader can see that in adverse atmospheric conditions VR needs other than its default parameters. Despite the number of unanswered questions our study raised we believe that the size of the two databases we used for our test experiments provide sufficient evidence that the virtual modalities can help deep NN classifiers improve performance, not only when inputs are acquired under marginal conditions, but also when all is, or looks, normal. Furthermore, we achieved this goal without changing their prototypical design of the NNs we use.

Overall, our present results suggest that the fusion of physical and derived artificial modalities can potentially improve the accuracy of AI systems.

(a) Raw Only (b) Trimodal (Raw.IN.VR)

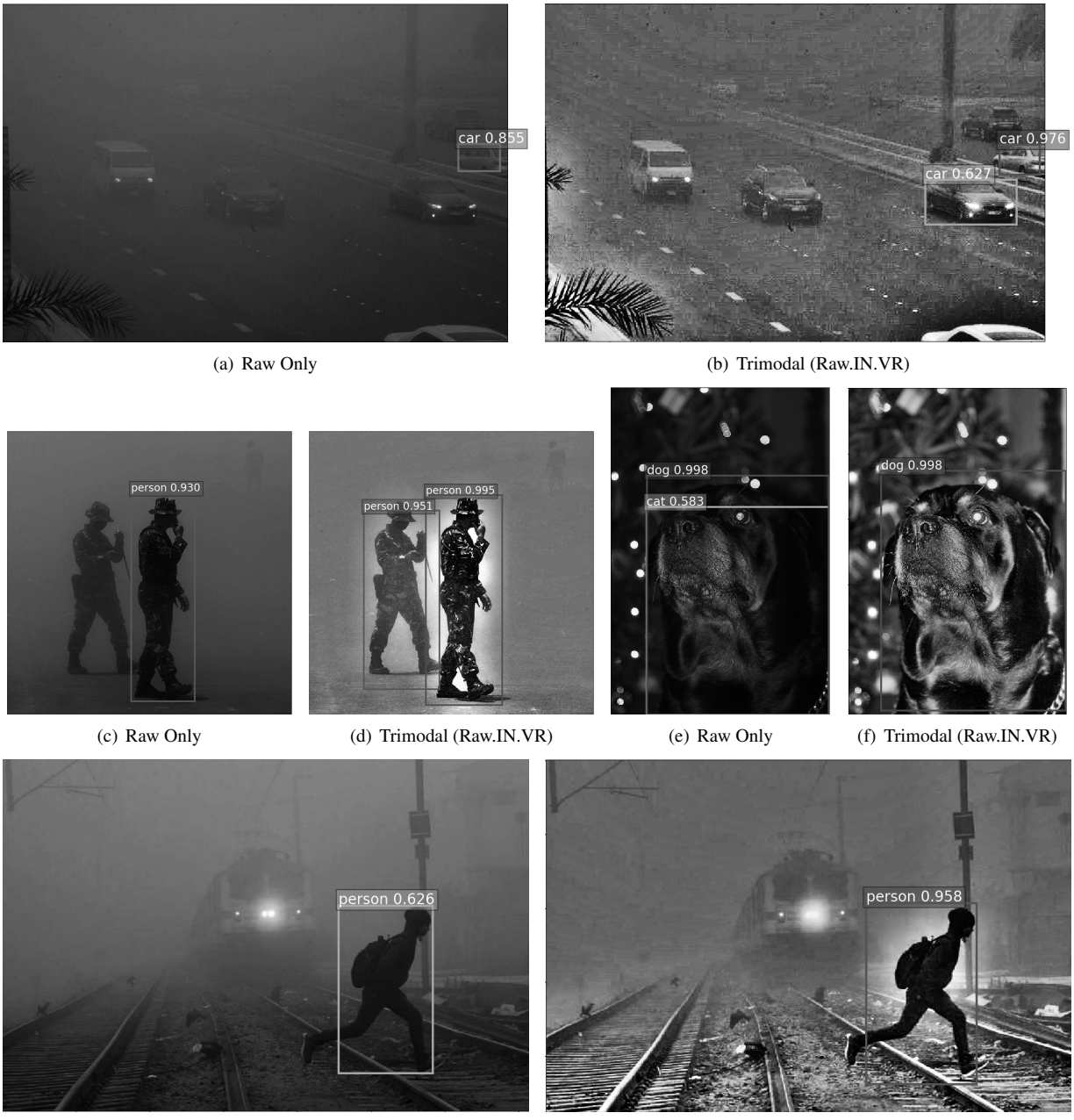
(c) Raw Only (d) Trimodal (Raw.IN.VR)

(e) Raw Only (f) Trimodal (Raw.IN.VR)

Figure 7: Each detection system draws a 'labeled' rectangle, enclosing in a tight manner any object (from the learned classes) they detect. The number on the side of the label represents the confidence, of the system, that the rectangle contains an object from the specific class (in the range $[0, 1]$, where 1 corresponds to certainty). In many occasions, the use of complementary information from the artificial IN and VR imaging modalities within our virtual multimodal Faster R-CNN architecture leads to higher detection capability (**1st column**), relatively to the unimodal system (**2nd column**) where only the raw image modality is used, especially when poor visibility obstructs the systems decision. System's detections with confidence below 0.4 are not displayed.

5 ACKNOWLEDGMENTS

This work was partially supported by NSF awards DMS-1720487 and DMS-1320910 and by a University of Houston Seed Funding for Advanced Computing (SeFAC), 2017 "Overcoming Poor Illumination for Real-time, Large Scale



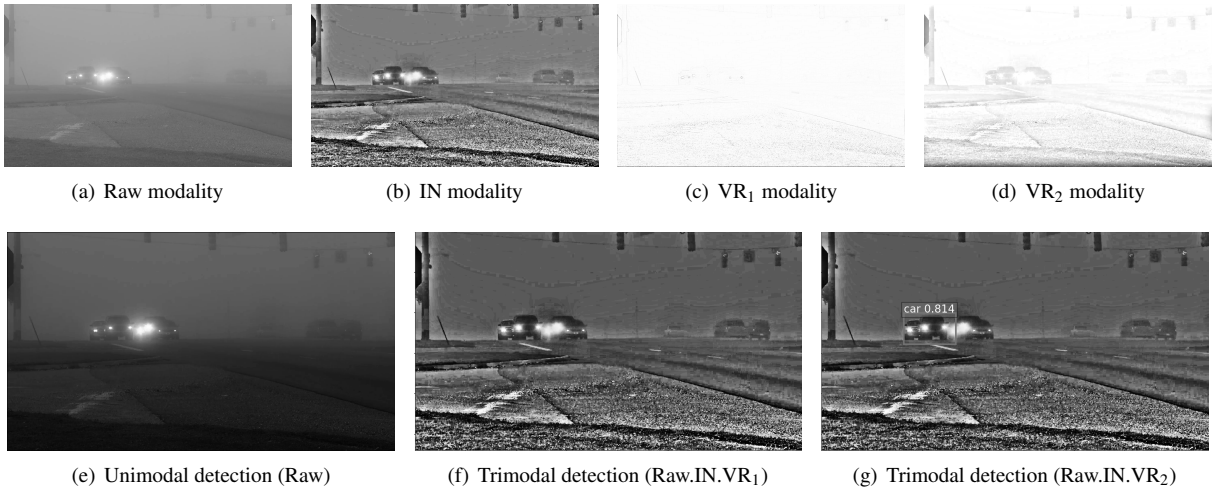
(a) Raw Only (b) Trimodal (Raw.IN.VR)

(c) Raw Only (d) Trimodal (Raw.IN.VR) (e) Raw Only (f) Trimodal (Raw.IN.VR)

(g) Raw Only (h) Trimodal (Raw.IN.VR)

Figure 8: As in Figure 7, system detections appear as 'labeled' rectangles, enclosing any object they detect. Again, the number on the side of the label represents system's confidence that the rectangle contains an object from the specific class (in the range $[0, 1]$, where 1 corresponds to certainty). Again we observe that the use of complementary information from the artificial IN and VR imaging modalities within our virtual multimodal Faster R-CNN architecture can lead to higher detection capability, relatively to the unimodal system where only the raw image modality is used. Higher detection capability of the Trimodal Raw.IN.VR system here translates into either correcting false detections of the unimodal Raw-only system (e.g. in (f)), or new accurate detections (e.g. in (d)), or improving significantly the system's confidence on accurate detections (e.g. in (h)).

Face Recognition Applications".



(a) Raw modality (b) IN modality (c) VR_1 modality (d) VR_2 modality

(e) Unimodal detection (Raw) (f) Trimodal detection (Raw.IN.VR_1) (g) Trimodal detection (Raw.IN.VR_2)

Figure 9: Optimizing the parameters of the image enhancement methods, used to derive the artificial imaging modalities, can further improve the multimodal system's accuracy. **Top row:** The raw (physical) modality (a) next to the derived (artificial) imaging modalities (b)-(d). Both (c) and (d) are generated by the VR method, but using different parameter settings. **Bottom row:** Improvement in detection performance, compared to the Raw-Unimodal Faster R-CNN detection attempt (e), is achieved only when Var_2 modality replaces Var_1 in the virtual Trimodal model (g).

References

Dalal, N. & Triggs, B. (2005), Histograms of oriented gradients for human detection, *in* 'Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01', CVPR '05, IEEE Computer Society, Washington, DC, USA, pp. 886–893.
URL: *http://dx.doi.org/10.1109/CVPR.2005.177*

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. & Fei-Fei, L. (2009), ImageNet: A Large-Scale Hierarchical Image Database, *in* 'CVPR09'.

Eitel, A., Springenberg, J. T., Spinello, L., Riedmiller, M. A. & Burgard, W. (2015), 'Multimodal deep learning for robust RGB-D object recognition', *CoRR* **abs/1507.06821**.
URL: *http://arxiv.org/abs/1507.06821*

Endsley, M. R. (1999), *Handbook of Aviation Human Factors*, Lawrence Erlbaum Asscoiates, chapter 11.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J. & Zisserman, A. (n.d.*a*), 'Pascal visual object classes challenge 2007 (voc2007) complete dataset'.
URL: *http://host.robots.ox.ac.uk/pascal/VOC/voc2007/*

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J. & Zisserman, A. (n.d.*b*), 'The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results', http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

Fu, X., Zeng, D., Huang, Y., Zhang, X.-P. & Ding, X. (2016), A weighted variational model for simultaneous reflectance and illumination estimation, *in* 'Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition', pp. 2782–2790.

Girshick, R. B. (2015), 'Fast R-CNN', *CoRR* **abs/1504.08083**.
URL: *http://arxiv.org/abs/1504.08083*

Girshick, R., Donahue, J., Darrell, T. & Malik, J. (2014), Rich feature hierarchies for accurate object detection and semantic segmentation, *in* 'Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition', CVPR '14, IEEE Computer Society, Washington, DC, USA, pp. 580–587.
URL: *https://doi.org/10.1109/CVPR.2014.81*

Johnsen, G., Ludvigsen, M., Sørensen, A. & Aas, L. M. S. (2016), 'The use of underwater hyperspectral imaging deployed on remotely operated vehicles-methods and applications', *IFAC-PapersOnLine* **49**(23), 476–481.

Land, E. H. (1964), 'The retinex', *American Scientist* **52**(2), 247–264.

Land, E. H. et al. (1977), *The retinex theory of color vision*, Citeseer.

Lin, T., Goyal, P., Girshick, R. B., He, K. & Dollár, P. (2017), 'Focal loss for dense object detection', *CoRR* **abs/1708.02002**.
URL: *http://arxiv.org/abs/1708.02002*

Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H. & Ng, A. Y. (2011), Multimodal deep learning., *in* L. Getoor & T. Scheffer, eds, 'ICML', Omnipress, pp. 689–696.

Ochoa-Villegas, M. A., Nolazco-Flores, J. A., Barron-Cano, O. & Kakadiaris, I. A. (2015), 'Addressing the illumination challenge in two-dimensional face recognition: a survey', *IET Computer Vision* **9**(6), 978–992.

Paul Viola, M. J. (2001), 'Rapid object detection using a boosted cascade of simple features', *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001.* **1**, I–I.

Redmon, J., Divvala, S., Girshick, R. & Farhadi, A. (2016), You only look once: Unified, real-time object detection, *in* 'The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)'.

Redmon, J., Divvala, S. K., Girshick, R. B. & Farhadi, A. (2015), 'You only look once: Unified, real-time object detection', *CoRR* **abs/1506.02640**.
URL: *http://arxiv.org/abs/1506.02640*

Redmon, J. & Farhadi, A. (2018), 'Yolov3: An incremental improvement', *CoRR* **abs/1804.02767**.
URL: *http://arxiv.org/abs/1804.02767*

Ren, S., He, K., Girshick, R. B. & Sun, J. (2015), 'Faster R-CNN: towards real-time object detection with region proposal networks', *CoRR* **abs/1506.01497**.
URL: *http://arxiv.org/abs/1506.01497*

Roper, W. E. & Dutta, S. (2006), Oil spill and pipeline condition assessment using remote sensing and data visualization management systems, *in* 'Freshwater Spills Symposium, Natural Disasters, Human Error, and Equipment Failure-Causes for Major Inland Oil Spills and the Resulting Multifaceted Response'.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R. & Lecun, Y. (2014), Overfeat: Integrated recognition, localization and detection using convolutional networks, *in* 'International Conference on Learning Representations (ICLR2014), CBLS, April 2014'.

Simon, M., Rodner, E. & Denzler, J. (2016), 'Imagenet pre-trained models with batch normalization', *arXiv preprint arXiv:1612.01452* .

Srivastava, N. & Salakhutdinov, R. R. (2012), Multimodal learning with deep boltzmann machines, *in* F. Pereira, C. J. C. Burges, L. Bottou & K. Q. Weinberger, eds, 'Advances in Neural Information Processing Systems 25', Curran Associates, Inc., pp. 2222–2230.
URL: *http://papers.nips.cc/paper/4683-multimodal-learning-with-deep-boltzmann-machines.pdf*

Sutskever, I., Martens, J., Dahl, G. & Hinton, G. (2013), On the importance of initialization and momentum in deep learning, *in* 'Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28', ICML'13, JMLR.org, pp. III–1139–III–1147.
URL: *http://dl.acm.org/citation.cfm?id=3042817.3043064*

Upadhyay, S., Mitsakos, N. & Papadakis, M. (2018), 'Real time visibility improvement for underwater video', *Proceedings of the 23rd Offshore Symposium, February 2018, Houston, Texas* .

Upadhyay, S. & Papadakis, M. (2019), A nonlinear multiscale method for illumination normalization. under revision, Applied Comp. Harmonic Analysis.