IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received November 6, 2020, accepted November 28, 2020, date of publication December 10, 2020,
date of current version December 28, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3043813

Pelee-Text++: A Tiny Neural Network for Scene Text Detection

**MANUEL CÓRDOVA[1], ALLAN PINTO[1], (Member, IEEE),
HELIO PEDRINI[1], (Senior Member, IEEE),
AND RICARDO DA SILVA TORRES[2], (Member, IEEE)**
[1]Institute of Computing, University of Campinas, Campinas 13083-852, Brazil
[2]Department of ICT and Natural Sciences, Faculty of Information Technology and Electrical Engineering, NTNU–Norwegian University of Science and Technology, 6009 Ålesund, Norway

Corresponding author: Ricardo da Silva Torres (ricardo.torres@ntnu.no)

This work was supported in part by the CNPq under Grant #307560/2016-3; in part by the São Paulo Research Foundation–FAPESP under Grant #2014/12236-1, Grant #2015/24494-8, Grant #2016/50250-1, Grant #2017/20945-0, Grant #2019/16253-1, Grant #2019/17729-0, and Grant #2019/22262-3; and in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior-Brasil (CAPES), Finance Code 001.

ABSTRACT Scene text detection has become an important field in the computer vision area due to the increasing number of applications. This is a very challenging problem as textual elements are commonly found in "noisy" and complex natural scenes. Another issue refers to the presence of texts encoded into different languages within the same image. State-of-the-art solutions rely on the use of deep neural network approaches or even ensembles of them. However, such solutions are associated with "heavy" models, which are computationally expensive in terms of memory and storage footprints, which hampers their use in real-time mobile applications. In this work, we introduce Pelee-Text++, a lightweight neural network architecture for multi-lingual multi-oriented scene text detection, especially tailored to running on devices with computational restrictions. Additionally, to the best of our knowledge, this is the first work to evaluate the performance of text detection methods in commercial smartphones. Over this scenario, Pelee-Text++ processes 2.94 frames per second and it is the only evaluated approach that did not cause memory issues on smartphones, even using an input image of 1024×1024 pixels. Our proposal achieves a promising trade-off between efficiency and effectiveness, with a model size of 27 Megabytes and F-measure of 91.20%, 85.78%, 81.72%, 80.30%, 82.53% and 66.51% on ICDAR 2011, ICDAR 2013, ICDAR 2015, MSRA-TD500, ReCTS 2019 and Multi-lingual 2019 datasets, respectively.

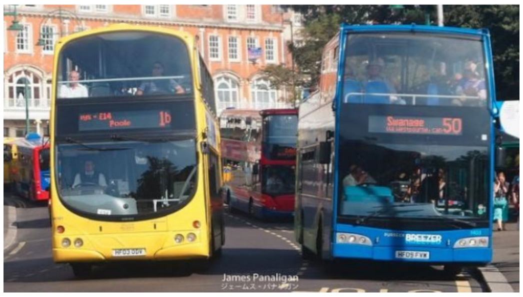
INDEX TERMS Text detection, mobile-network, mobile devices, multi-oriented text, multi-lingual, convolutional neural network.

I. INTRODUCTION

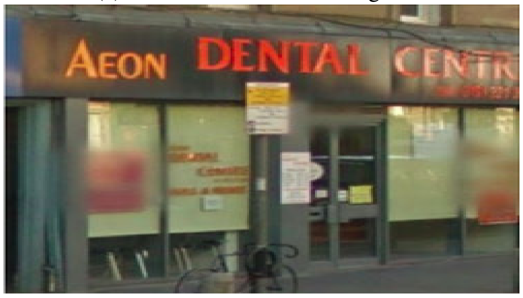
Nowadays, the detection and recognition of scene texts have become important topics in machine learning and computer vision areas due to the daily use of digital cameras and the huge amount of applications related to this field, such as mobile and context-aware services [60], traffic sign detection [10], [11], image retrieval [58], blind person assistance [61], and text translation [53]. In fact, new applications are still emerging, for example, those related to the interpretation of scene textual content (Figure 1). However, both scene text detection and recognition are more challenging than traditional document processing given the presence of different types of text and scenarios related to complex/natural backgrounds, font styles and sizes, blurring, orientations, occlusion, aspect ratios, perspective projections, among others. Despite the application, the text detection task plays an important role in the final recognition result since this task precedes the recognition step. Thus, the development of a good text detector is paramount to reach efficient and effective scene text recognition systems.

Addressing text detection is difficult since text images have different visual properties depending on their source, for example, born-to-digital (e.g., e-mails, advertisements, Web images) or incidental/focused scene text (scene text images taken from wearable cameras or urban captures). Additionally, text could appear with arbitrary-orientations, perspective distortion, and be associated with even more challenging scenarios, for example, those related to the presence of different languages in the same scene. In this regard, several public

The associate editor coordinating the review of this manuscript and approving it for publication was Ahmed Farouk.



(a) Where does the blue bus go to?



(b) What kind of business is this?

FIGURE 1. Scene Text Visual Question Answering [2].

datasets have been built to foster the creation of solutions to overcome these problems, such as SynthText [12], ICDAR 2011 [45], ICDAR 2013 [24], ICDAR 2015 [25], ReCTS 2019 [64], Multi-lingual 2019 [40], and MSRA-TD500 [59].

Given the complexity of the text detection problems, sophisticated supervised approaches have been used in the state-of-the-art solutions. Moreover, the detection quality impacts directly on the recognition result, i.e., whether a predicted bounding box covers a entirely word or just part of it impacts on the effectiveness of recognition algorithms.

Despite the huge amount of methods developed to text detection proposed in the last years, most of these studies are focused on improving their results without concerns about their model size. Those approaches use VGG-16 [1], [4], [8], [29], [36], [65], [68] and ResNet [31], [32], [34], [38], [52], [63] as feature extractors, producing models with a size ranging from 80 Megabytes [1] to more than 350 Megabytes [38]. For this reason, in practice, their use on devices with computational constraints turns non-viable.

Recently, the edge computing concept has empowered the next generation of machine learning applications [26]. Conversely to cloud computing, edge computing is revolutionizing the way embedded systems are architected by moving complex processing and analysis to end devices (e.g., mobile and wearable devices). Cloud services brought several advantages for machine learning, such as fast computation and almost unlimited storage; however, its throughput and response time is not enough to ensure its use in real-time applications, which are also impacted by the latency fluctuation in wide-area networks. Furthermore, one of the biggest concerns in mobile devices is the energy consumption, and data transferring over the network implies more energy [5], [26].

The use of tiny neural network models, as a part of fully deployed mobile applications, has some advantages for real-time applications [13], [26], such as: (a) efficiency, in terms of time processing, is one of the most important considerations for real-time mobile applications; (b) local processing, even with the huge amount of online services, some bottlenecks on cloud services or fluctuations on network latency could have a big impact on the performance of real-time applications, and even the privacy leaks could increase; (c) energy consumption is also affected when large deep learning models are used.

Related to text detection, there exist few works using lightweight mobile-oriented neural networks [6], [7], [9]. Some of the approaches based on tiny neural network models [7], [9] are limited to the detection of horizontal text. In this vein, our previous work, Pelee-Text [6], introduced a light architecture for multi-oriented scene text detection, reaching competitive results in several datasets with a model size of 40 Megabytes.

Herein, based on Pelee-text [6], we came up with Pelee-Text++. Pelee-Text++ is the result of a comprehensive study of Pelee-Text. We evaluated the impact on the efficiency and effectiveness of each one of its main components: (i) evaluation of the different convolutional blocks both in behavior and number, (ii) influence of aspect ratios, (iii) impact of different scales of input images. As a result, Pelee-Text++ is an even more compact and simpler neural network architecture for multi-lingual multi-oriented scene text detection suitable for running on devices with computational constraints.

Pelee-Text [6] and Pelee-Text++ are compact neural networks for text detection that have reached a good trade-off between efficiency and effectiveness, becoming promising approaches to mobile applications. Both use the same kind of convolutional blocks: (i) stem block, (ii) transitional blocks, and (iii) dense blocks. However, the main difference between them is the structure of their backbones. Pelee-Text has a total of 27 blocks (1 stem-block, 5 transitional-blocks, and 19 dense-blocks), while Pelee-Text++ has been compressed to 14 blocks (1 stem-block, 5 transitional-blocks, and 8 dense-blocks). Additionally, Pelee-Text++ uses 1, 2, 3, 1/2, and 1/3 as aspect ratios, cutting-off the 5 and 1/5 aspect ratios used by Pelee-Text. The new backbone of Pelee-Text++ along with the use of less aspect ratios brought a huge impact on the efficiency and compacting even more the model size.

Regarding efficiency and effectiveness, in most of the scenarios, Pelee-Text++ outperforms the results of our previous work. Moreover, in a real mobile scenario and considering the best setup (input image of 300×300 pixels), Pelee-Text++ is $1.46\times$ faster than Pelee-Text and, in the worst case (input image of 1024×1024 pixels), Pelee-Text++ was just 41.50% of the processing time of Pelee-Text. Finally, in terms of model size, whereas Pelee-Text has a weight of 40MB, Pelee-Text++ is only 27MB.

Our new proposal was evaluated over six public available datasets: ICDAR 2011 [45], ICDAR 2013 [24], ICDAR 2015 [25], MSRA-TD500 [59], ReCTS 2019 [64], and Multi-lingual 2019 [40] obtaining competitive results against state-of-the-art methods. Experimentally, our proposal demonstrated its ability to work over scenarios with different particularities. Pelee-Tex++ is at least 2.96 times smaller than state-of-the-art methods, with a processing time of 23.25, 15.06, and 3.65 FPS, for its 768, 1024, and multi-scale versions.

Furthermore, to the best of our knowledge, this is the first study that evaluates the efficiency of several text detection methods in a real mobile scenario. For this, we assessed their performance in smartphones using four different scales of input images. These experiments showed the drawbacks of state-of-the-art methods when a real mobile environment is used. On mobile devices, our proposal is capable of processing 2.94 FPS, being at least 5.5 times faster than CRAFT [1], which is one of the best methods in several datasets with a model size of 80 Megabytes.

Our contributions can be summarized as:

1) Design of a lightweight neural network architecture to detect multi-lingual multi-oriented scene texts.
2) Evaluation of text detection methods in devices with computational constraints, i.e., performance of well-known text detection methods in commercial smartphones.

The remainder of the paper is organized as follows. In Section II, we provide an overview of related work. Our method, Pelee-Text++, is presented in Section III. Section IV details the adopted experimental protocol. Section V presents and discusses achieved results. Finally, Section VI presents the conclusions and future work.

II. RELATED WORK

A. SCENE TEXT DETECTION

During the last years, Convolutional Neural Networks (CNN) have become promising approaches to deal with several challenging scenarios in scene text localization, such as multi-scale detection [14], oriented text detection [29], text detection in complex backgrounds [12], and arbitrary-shaped text [36].

The proposed approaches have used different techniques for dealing with scene text detection, most of them are focused on regression [6], [28], [31], [51], [57], segmentation [4], [8], [56], [56], or the combination of both techniques [35], [38], [39], [52], [63]. Additionally, some proposals have merged the detection and recognition tasks as a jointed pipeline for improving their results [34], [38], or had even used knowledge distillation [68].

In the vein of regression methods, generally, this type of approaches generate two points $(x_{\min}, y_{\min}, x_{\max}, y_{\max})$ [28], four points $(x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4)$ [6], [18], [29], [63], or adaptive number of points [51] as a bounding box prediction. Furthermore, they predict offsets from predefined anchor-boxes [6], [28], [29], direct regression for predicting offsets from a single point [18], or even shape regression [57].

On the other hand, text instance segmentation approaches have been working on pixel-wise classification for defining neighborhood linkages [8], salient maps [67], even proposing progressive scale expansion algorithm [50] to improve the separation between near text instances.

Recently, some approaches used several branches with the goal of taking advantage of the fusion of bounding box regression and text instance segmentation techniques [32], [35], [39], [52], [63]. Most of those approaches are inspired by a well-known object detector, Mask-RCNN [16]. In a general way, these methods [18], [31] use a multi-scale feature extraction based on Feature Pyramid Networks (FPN) [30], or even 3D special pyramid mask for a better characterization of text instances [32]. Next, these features are fed to several branches, where each one of them has specific tasks, such as bounding box regression, text instance segmentation or refinement modules.

Other researches have adopted different text detection strategies. CRAFT [1] and WeText [48], for example, works with character-level annotations. CRAFT [1] predicts a region character score using a Gaussian heatmap and an affinity score of neighboring characters, while WeText [48] proposed a weakly-supervised approach to text detection that works over non-annotated data or weakly annotated data, and a graph based method is used to define the final results.

For its part, DRRG [65] dealt with text detection using a graph convolutional network. First, local graphs are created based on the linkage between text components. Then, a deep relational linkage is performed based on the previous discovered local graphs and a Breath First Search is applied to join the linkages for final prediction. Attention maps have also been used in this area. For instance, GISCA [4] improved text characterization based on Contextual Attention Module (CAM) and a Gradient-Inductive Module (GIM).

Unlike previous works, some researches have fused detection and recognition tasks. In this type of approaches, the recognition branch fed the detection one for filtering false positives in an end-to-end scheme [27], [34], [38]. Additionally, some approaches used data augmentation techniques to improve the results and the generalization of text detection methods. In this context, generative models using domain shifts from cross-domain [62] and sampling of sub-regions of text segments through boostrapping [56] were proposed.

Current solutions have addressed text detection challenges by using deep architectures, such as VGG [46] and ResNet [15], or even ensembles of them, producing models with a size ranging from 80 Megabytes [1] to more than 350 Megabytes [38]. Such solutions are computationally expensive, which makes them unfeasible, in practice, to be used in devices with computational constraints, such as memory, computational power, bandwidth and energy [13]. In this regard, some "mobile" CNN architectures have been already proposed [19], [22], [44], [49], [66], i.e., lightweight convolutional neural network architectures specifically designed for mobile devices.

Based on MobileNetv2 [44], MobText [7] is the state of the art on ICDAR 2011 [45] with a model size of 37 Megabytes. Based on the same mobile architecture, Fu et al. [9] proposed a neural network of just 16 Megabytes inspired by a U-Net approach [43]. Nevertheless, these two methods were just evaluated on datasets with horizontal text, since they are based on the typical rectangular bounding box representation, which are not able to capture oriented text.

In the same vein, Xue *et al.* [37] proposed OctShufle, which uses a combination of ResNet blocks [15] and Shuffle units [66] to produce a model size of 88.79 Megabytes. However, it has problems on detecting oriented text. Finally, Pelee-Text [6], which uses PeleeNet [49] as feature extractor, appeared as a mobile oriented architecture for multi-lingual scene text detection with a model size of 40 Megabytes and promising results on several datasets. Pelee-Text provides a promising trade-off between effectiveness and model size.

B. LIGHTWEIGHT NEURAL NETWORKS

In several tasks of computer vision, state-of-the-art approaches have used deep convolutional neural networks to improve results. However, these approaches tend to go deeper, increasing the number of parameters without concerns on the amount of operations and the final model size. For this reason, it is difficult to use these approaches on devices with computational constraints. In order to overcome this problem, some works have focused their efforts on the proposal of lightweight neural networks [22], [42], [44], [66].

MobileNets [19], [44] were proposed as a computational time and model size efficient neural networks for mobile applications. These approaches are based on depthwise separable convolution, where fully convolutional layers are split into two in order to reduce computational time and the number of parameters without greatly affecting performance. Similarly, ShuffleNet [66] applies a grouping approach to divide channels through a point-wise convolution. In addition, this approach uses cross-group information flow.

SqueezeNet [23] is another approach focused on reducing the number of parameters with the goal of compressing the final model. SqueezeNet replaces most of the 3×3 filters with 1×1 filters and reduces the amount of input channels to 3×3 filters. In the same vein, Yolo-Lite [22] is a real-time object detector for non-GPU computers, which is a simplified version of the network proposed by Yi and Tian [41]. This is a very compact network that uses just 7 convolutional layers and does not use batch normalization.

Based on a neural architecture search (NAS), Yi *et al.* [69] proposed NASNet as a compact neural network model. Instead of looking for a complete architecture, the authors reduced the problem to find the best convolutional layer architecture. Then, the final neural network is built with exactly this type of convolutional layers.

Taking advantage of the properties of well-known networks [15], [20], [47], PeleeNet was proposed as a feature extractor. This method uses a stem block to improve the feature space of the input, two-way-dense layers with different filters with the goal of extract relevant features for large objects, and its bottleneck layer adapts the input dimension dynamically. Furthermore, the authors built Pelee for object detection using PeleeNet along with an optimized version of SSD [33].

Several experiments have demonstrated that PeleeNet is a very efficient and effective network for scenarios with computational constraints. In terms of millions (M) of parameters, PeleeNet has 5.3M of parameters, using 2.5M and 2.4M less than NASNet-A [69] and ShuffleNet2x [66], respectively. In addition, PeleeNet is 66% the size of the MobileNet model and 11.3× smaller than YOLOv2 [41]. On NVIDIA TX2, PeleeNet is faster than MobileNet [19], MobileNetv2 [44] and ShuffleNet2x [66], especially when a half-precision floating-point is used. Complementary, for object detection, Pelee achieved comparable or better results than MobileSSD [21], Yolov2 [41] and ShuffleNet [66].

III. PROPOSED METHOD: PELEE-Text++

This section presents our proposal, a lightweight convolutional neural network architecture for multi-oriented multi-lingual scene text detection. More specifically, our goal is to introduce a competitive and efficient approach more appropriate for devices with computational constraints.

A. OVERVIEW

This study presents Pelee-Text++, which reflects our effort towards the design of a tiny neural network based on recent works that rely on mobile-oriented architectures, originally proposed for object detection. More precisely, the solution introduced in this section is an extension of our previous work [6], named Pelee-Text. Our architecture is based on PeleeNet [49] and TextBoxes++ [29] networks. By taking advantage of their best particularities, we propose a faster and lighter architecture for detecting scene text, becoming a more viable solution to be used in constrained processing devices such as smartphones and tablets.

PeleeNet was proposed by Xu *et al.* [49] as a novel neural network architecture for mobile devices. It is a variant of DenseNet [20] and its main goal was to work on strict memory and computational constraints. Furthermore, PeleeNet along with an optimized version of Single Shot MultiBox Detector (SSD) [33] was proposed for object detection. Unlike original SSD, it does not use the 38×38 feature map; but two distinct scales of prior-boxes over 19×19 feature map instead.

Several experiments, demonstrated that PeleeNet is a very efficient and effective network for scenarios with computational constraints. In terms of millions (M) of parameters, PeleeNet has 5.3M of parameters, using 2.5M and 2.4M less than NASNet-A [69] and ShuffleNet2x [66], respectively. Moreover, PeleeNet is 66% the size of the MobileNet model and 11.3× smaller than YOLOv2 [41]. On NVIDIA TX2, PeleeNet is faster than MobileNet [19], MobileNetv2 [44],

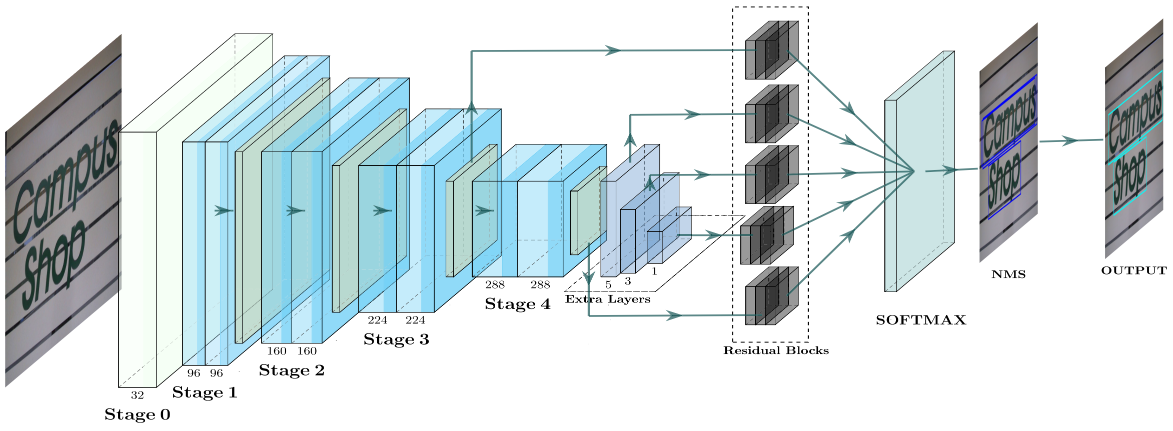


FIGURE 2. **Overview of Pelee-Text++ architecture.**

and ShuffleNet2x [66] when half-precision floating-point is used. In a complementary manner, for object detection, Pelee achieved comparable or superior results to MobileSSD [21], Yolov2 [41], and ShuffleNet [66].

Regarding to a specific bounding box regression approach for text detection, He *et al.* [29] proposed TextBoxes++ as an end-to-end convolutional neural network to detect arbitrary-oriented word bounding boxes. To predict the regions in the image that contain texts, the authors used VGG-16 as feature extractor and a modified SSD adapting some layers in order to detect text with longer aspect ratios. At the end, a non-maximum suppression (NMS) procedure is applied to filter the final outputs.

As a result of combining the best of both architectures, we propose a tiny neural network specifically designed for scene text detection. In natural scenes, multiple challenging scenarios emerge, such as different font styles, blurring, orientations, image projections, among others. With the presence of text with different orientations and particular projections, the typical rectangular bounding boxes are not enough, being necessary the use of quadrilaterals.

For that reason, our network predicts quadrilaterals with points $P_n = (x_n, y_n)$ in clockwise order (P_1, P_2, P_3, P_4), being P_1 the top left point. Each one of the predicted quadrilaterals is evaluated as containing text or background. Moreover, for a covering most text regions, we dense prior-boxes based on vertical offsets. Furthermore, we used a simplified version of SSD [33] with different scales of bounding boxes over the 19×19, 10×10, 5×5, 3×3, and 1×1 feature maps.

B. ARCHITECTURE

An overview of our architecture is presented in Figure 2. Our feature extractor is composed of five stages. In Stage 0, we improve the characterization by increasing the number of channels of the input image. Then, Stages 1 to 4 are based on two blocks of Two-Way Dense Layers with 3×3 convolutions, which have the goal of look for useful features to describe text regions. Unlike DenseNet [20] and inspired in PeleeNet, our network manages the channel expansion with two convolutional paths each one working with half of the channels. Finally, a transitional convolutional block keeps the discriminability of the features without impact the number of channels between stages.

Furthermore, extra convolutional blocks from SSD [33] corresponding to the 5×5, 3×3, and 1×1 feature maps are added after Stage 4. Pelee-Text has six text-specific layers designed to look and define final bounding boxes. These layers are built considering 3×5 kernels, given than text covers long continuous regions, becoming powerful to detect text that usually has longer aspect ratio than traditional object detection, and also with the presence of textual elements with some orientation and/or projection. They perform bounding box prediction and binary bounding box classification (text or background) at the same time $(x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4, \textit{confidence})$.

Our method is based on the regression of offsets taking, as a starting point, a set of prior-boxes predefined for each one of the six layers. For these, before predictions, outputs from five layers pass through residual blocks. Those layers are the last layer of Stages 3 and 4 along with the three layers of the simplified version of SSD. The feature map from the last layer of Stage 3 is used twice using different scales of prior-boxes.

Furthermore, we used some aspect ratios (1, 2, 1/2, 3 and 1/3) with the goal of increasing the amount of default prior-boxes. Applying these aspect ratios, we produce larger horizontal boxes, and more important, vertical prior-boxes are generated. This initial set of prior-boxes is the base for the regression of multi-oriented quadrilaterals. Additionally, for vertical crowded text regions, we increase the number of prior-boxes applying a dense vertical distribution in order to minimize the number of text elements lost in this type of scenario. For guiding the training, we use the loss function from [33], which involves the confidence (L_{conf}) and localization (L_{loc}) losses:

$$L(p, c, l, g) = \frac{1}{N}(L_{conf}(p, c) + \alpha L_{loc}(p, l, g)), \qquad (1)$$

where p are the estimated bounding boxes, c is the confidence of being text, l is the predicted localization, g corresponds to the ground truth, N is the number of matched boxes, and α is the weight for the L_{loc}. Furthermore, we used the smooth L1 loss for L_{loc} and the soft-max loss for L_{conf}.

Additionally, during testing, we exploit a multi-scale procedure [6], [29], [34], which uses input images with four different sizes: 384, 768, 1024, and 1536. In Table 1, we can see the heatmaps produced by the bounding box source layers with those four scales as inputs. Each one of these feature maps captures complementary information about textual elements of different sizes, and exploits the relation between different scales and bounding box sizes. The smaller scales (384, 768) allow to detect larger objects, while the larger scales (1024, 1536) are able to capture small textual elements. At the end, before presenting final predictions, a cascade NMS based on the Intersection over Union (IOU) is performed to discard overlapped bounding boxes from the four scales using an IOU ≥ 0.1.

Currently, neural networks are state-of-the-art solutions for scene text detection. However, their solutions are more concerned with effectiveness than efficiency, some works even fuse two or more deep architectures for improving results. In terms of Megabytes (MB), they produce models ranging from 80MB [1] to more than 350MB [38]. In this vein, one of the main points of our proposal is its compact architecture with a weight of only 27MB and 7 millions of parameters, being at least 2.96× smaller than its counterparts, becoming a very promising architecture for mobile applications.

C. ABLATION STUDY

The proposed architecture was built as a result of an ablation study considering our previous approach, Pelee-Text [6]. The objective is to discover the impact of each component and propose our new lighter neural network architecture for scene text detection. Our main concern is to improve its efficiency without hurting too much its effectiveness. In this vein, the impact of four main components are evaluated: (i) stem block responsible for improving the characterization of the image, (ii) dense blocks with the goal of looking for useful features, (iii) transitional blocks that keep the discriminability between stages, and (iv) residual blocks to improve the feature representation before the bounding boxes prediction. Therefore, we cut it off one component at the time to evaluate its influence on our network (See Table 2).

As result, we could see that Stem and Residual blocks are very important as part of the feature extractor, and how the number of dense blocks in the different stages could influence the results and model size of our model. Finally, we selected the architecture built in the sixth setup based on the trade-off between F-measure and model size compared to our baseline, Pelee-Text [6]. Our final architecture has a model size of 27 Megabytes reaching a model compression of 32.5% compared to the baseline with a drop of just 0.38 percentage points on F-measure. More important, as we will explain later, this architecture brings a huge advantage compared to Pelee-Text when a real mobile environment is used to evaluate its performance.

IV. EXPERIMENTAL SETUP

In this section, we describe the datasets, metrics, and protocols used to asses the effectiveness of our method and its counterparts.

A. DATASETS AND METRICS

Initially, for performing experiments and evaluating our models, we used datasets and metrics already available and widely used in the literature. Such datasets present different particularities. We evaluated our proposal over six well-known public available datasets: ICDAR 2011 [45], ICDAR 2013 [24], ICDAR 2015 [25], ReCTS [64], Multi-lingual 2019 (MLT) [40], and MSRA-TD500 [59].

The SynthText [12] and MLT 2019 datasets were used for pre-training since some datasets have few images for training. Table 3 shows details about each dataset, such as the number of images, text-orientation, and languages that appear in their images.

Regarding to the metrics, the effectiveness of our method is measured in terms of Recall (R), Precision (P), and F-measure (F1). More precisely, we evaluated our results using the evaluation tool public available by the International Conference on Document Analysis and Recognition (ICDAR)[1] in each one of its competitions. For ICDAR 2013 dataset, we used the ICDAR13 metric, whereas for the MSRA-TD500 dataset, its ground truth is represented in quadrilateral format (P_1, P_2, P_3, P_4). Furthermore, the efficiency takes into account the processed Frames per Second (FPS), as well as memory and storage footprints.

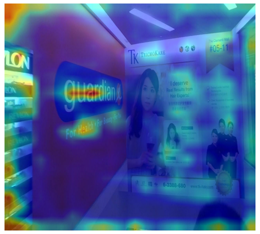
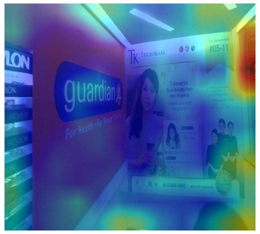
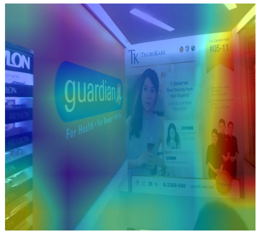
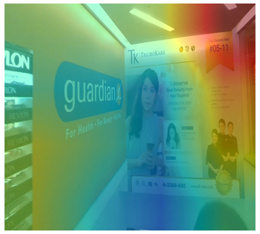
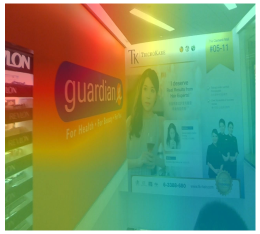
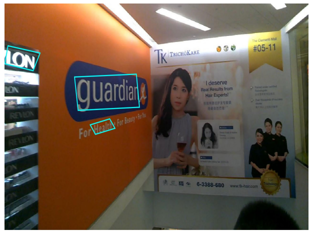
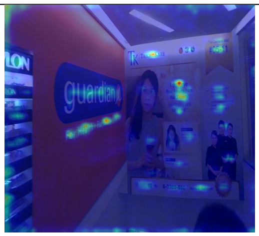
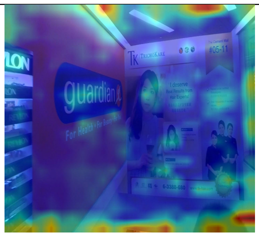
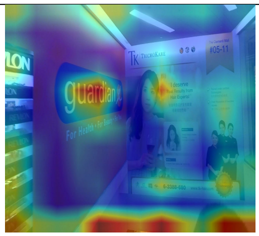
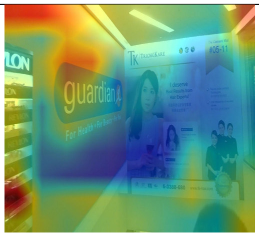
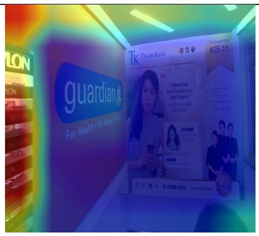
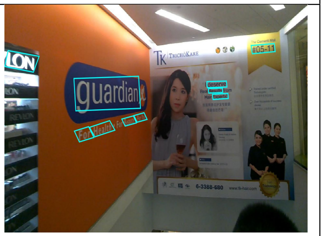
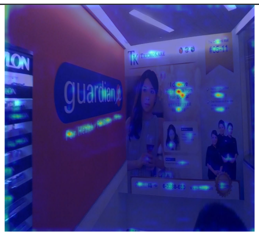
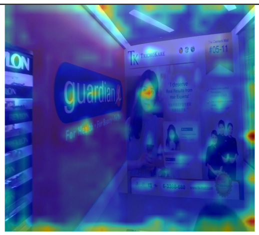
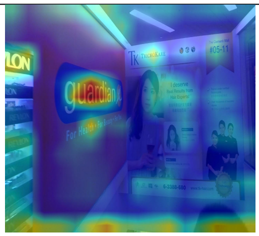
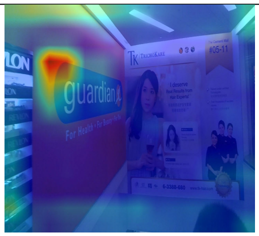
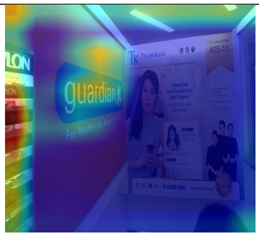
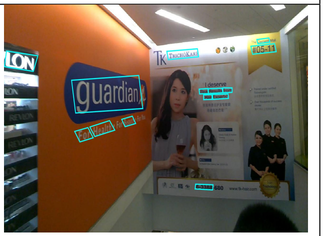
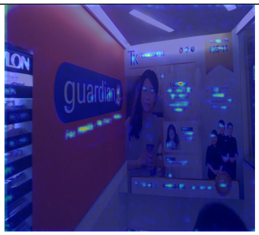
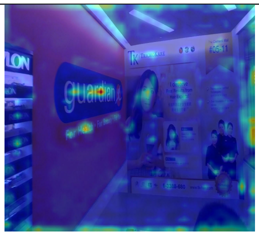
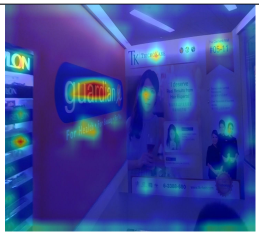
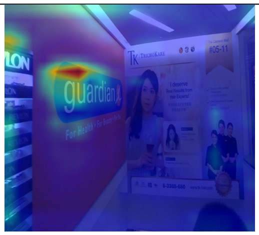
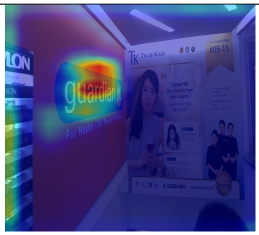
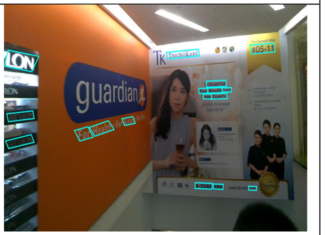
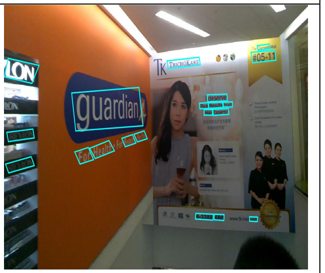
B. TRAINING PROTOCOL

For the experiments, we followed a four-stage training scheme where difficult cases were not considered, i.e., text cases with transcription "###" in the ground truth. First, we pre-trained our models in two stages using SynthText (Stage 1) and MLT 2019 (Stage 2) datasets with an input image size of 384, then two fine-tuning stages were performed on each dataset with 384 and 768 as input image size, preparing our network for multi-scale detection. For 384 and 768 training stages, we used a batch size of 48 and 20, respectively. Parameter values, such as learning rate, negative ratio, L_{loc} weight (α), and weight decay, were defined empirically through several experiments in the training set.

We pre-trained our models during 10 epochs on SynthText and 200 epochs on MLT 2019. Moreover, we used the

[1]https://rrc.cvc.uab.es (As of July 2020).

TABLE 1. Heatmaps: layers and scales.

Scales	Bounding Box Source Layers					Detection results
	Stage 3	Stage 4	SSD 5 × 5	SSD 3 × 3	SSD 1 × 1	
384						
768						
1024						
1536						
					Multi-scale Detection	

Stochastic Gradient Descent (SGD) to optimize our network and the "Xavier" technique for initializing the weights. The learning rate, negative ratio and L_{loc} weight (α), weight decay, and momentum were 5×10^{-3}, 3, 0.8, 5×10^{-4}, and 0.9, respectively. In Stage 3, we trained for about 300 epochs over almost all the datasets, except for the MSRA-TD500 given that the bounding boxes of this dataset present the particularity of covering text lines instead of words, so we trained for 600 epochs.

Additionally, depending on the stage and dataset, we used different values for parameters, such as learning rate, steps for learning rate decay, L_{loc} weight, ratio between the negatives and positives, and number of iterations. Table 4 shows the parameter values used during the fine-tuning stages.

For testing, we discarded the predictions with a detection score less than 0.5, 0.6, 0.5, 0.8, 0.6, and 0.4 for ICDAR 2011, ICDAR 2013, ICDAR 2015, MSRA-TD500, ReCTS, and MLT. At the end, NMS was applied with an overlap threshold of 0.1 for all datasets.

All experiments were conducted on an Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz with 12 cores, 64GB of RAM, Ubuntu 64-bits OS and two GeForce GTX 1080ti.

C. MOBILE PROTOCOL

In order to evaluate the efficiency of our proposal against some of the state-of-the-art methods in mobile devices, we executed experiments over three smartphones: Motorola Moto G6 Play, Asus Zenfone 5, and Xiaomi Mi 9T. These smartphones have different specifications which are detailed in Table 5. As we can see, these smartphones have different RAM Memory capacity ranging from 3GB to 6GB, and with different kinds of processors and clock speed (from 1.4GHz to 2.2GHz).

Moreover, to assess the efficiency of each method, we used the test set of ICDAR 2015 that contains 500 images with a variety of natural scenes. We computed the mean time on the whole test set, and we used four different scales of images (300, 384, 768, and 1024) to evaluate memory footprint and performance in each one of these scenarios.

For this, we built a basic Android app for running different neural network models. Our implementation was based on three Github repositories corresponding to Caffe,[2] PyTorch,[3] and Tensorflow [4] implementations.

V. RESULTS AND DISCUSSION

This section presents the results of our proposal against several state-of-the-art methods, considering three setups: results on datasets containing English text (Section V-A), multilingual text (Section V-B), and experiments related to efficiency aspects on mobile devices (Section V-C). Additionally, we also collected information about the model size and/or number of parameters of the baseline methods. That information was taken from their respective papers or authors' official Github.

In order to deal with a more specific field, some approaches to text detection are adapted or inspired by well-known object detectors [38]. In the vein of mobile-oriented neural network architectures, Liu *et al.* [7] proposed MobText as a text detector approach based on MobilenetV2 [44]. Their proposal achieved the best trade-off between efficiency and efficacy compared to Yolov3 [42] and SqueezeDet [54]. For this reason, we used MobText [7] as a baseline to compare our approach against mobile-oriented text detection methods.

A. DETECTING ENGLISH TEXT

We evaluated our proposal in three scenarios containing English text, i.e., born-digital images (ICDAR 2011), natural scenes with horizontal or near horizontal text (ICDAR 2013), and natural scenes with arbitrary-oriented text (ICDAR 2015).

First, we evaluated our proposal on the ICDAR 2011, which contains digital-created images that have some JPEG artifacts making this dataset a very different scenario compared to datasets collected from natural scenes. As we can see in Table 6, Pelee-Text++_MS obtained a F-measure of 91.20% outperforming most of the methods, even our previous version, Pelee-Text, and it is only placed behind MobText [7] which is the state-of-the-art on this set of images.

On ICDAR 2013, a dataset from natural scene images with presence of horizontal or near horizontal text, Pelee-Text++ reached competitive results with F-measure of 79.72% and 85.78% for its 768 and multi-scale versions, defeating Pelee-Text with a model 13 Megabytes lighter (see Table 7). Despite having a lower F-measure compared to state-of-the-art methods, such as CRAFT [1], MaskTextSpotter [38], PMTD [32] and FTPN [31]; our proposal obtained a good trade-off between efficacy and model size. Figure 4a shows the balance between effectiveness and model size compared to state-of-the-art methods.

Compared to the best methods, our proposal is 12.96× smaller than MaskTextSpotter [38] and 2.96× than CRAFT [1]. On the other hand, focusing on methods with light models, MobileNetv2+UNet [9] (16MB) and MobText [7] (37MB) have F-measures with 9.78 percentage points lower than Pelee-Text++ (27MB). Although Pelee-Text++ has a competitive result, we have to work on the missing cases with special focus on the detection of text in images with the presence of blurring and occlusion (see Figure 3b).

ICDAR 2015 is a dataset that came up with new variants of text orientations in natural scenes, i.e., vertical text, and text with visual projections. This dataset works over quadrilaterals instead of the typical rectangular bounding boxes. For that reason, MobileNetv2+UNet and MobText has not been tested on this dataset. As these methods use a typical rectangular representation, they are not tailored to the localization of oriented text. FCN [67] (57MB) and OctMLT [37]

[2]Caffe: https://github.com/sh1r0/caffe-android-lib (As of July 2020).

[3]Pytorch: https://github.com/pytorch/android-demo-app (As of July 2020).

[4]Tensorflow: https://github.com/tensorflow/.../examples/android (As of July 2020).

TABLE 2. Ablation study using ICDAR 2015.

	Components				Aspect		Model Size	# Parameters
	Stem	Transitional	Residual	Dense	Ratio	F-measure	(MB)	(Millions)
1	✓	✓	✓	[3, 4, 8, 6]	[2, 3, 5]	79.46	40	10.35
2		✓	✓	[3, 4, 8, 6]	[2, 3, 5]	77.99	40	10.34
3	✓		✓	[3, 4, 8, 6]	[2, 3, 5]	79.87	37	9.51
4	✓	✓		[3, 4, 8, 6]	[2, 3, 5]	77.43	46	11.91
5	✓	✓	✓	[1, 4, 8, 6]	[2, 3, 5]	79.22	39	9.95
				[3, 2, 8, 6]	[2, 3, 5]	80.27	39	9.95
				[3, 4, 4, 6]	[2, 3, 5]	80.70	37	9.54
				[3, 4, 8, 2]	[2, 3, 5]	78.92	38	9.70
				[1, 2, 2, 2]	[2, 3, 5]	77.80	32	8.20
				[2, 2, 2, 2]	[2, 3, 5]	79.50	32	8.30
6	✓	✓	✓	[2, 2, 2, 2]	[2, 3]	**79.08**	**27**	**7.01**
7	✓	✓	✓	[2, 2, 2, 2]	[2, 5]	**78.27**	**27**	**7.01**
8	✓	✓	✓	[2, 2, 2, 2]	[3, 5]	**77.96**	**27**	**7.01**

TABLE 3. Datasets used in our experiments.

Dataset	# Train Images	# Test Images	Text Orientation	Languages
SynthText [12]	858, 750	–	Horizontal	English
ICDAR 2011 [45]	410	141	Horizontal	English
ICDAR 2013 [24]	229	233	Horizontal	English
ICDAR 2015 [25]	1000	500	Arbitrary-Oriented	English
MSRA-TD500 [59]	300	200	Multi-Oriented	English, Chinese
ReCTS 2019 [64]	20000	5000	Multi-Oriented	English, Chinese
MLT 2019 [40]	10000	10000	Arbitrary-Oriented	Arabic, English, French, Chinese, German, Korean, Japanese, Italian, Bangla, Hindi

TABLE 4. Training protocol values.

Datasets	Stage 3						Stage 4					
	Image Size	Negative Ratio	L_{loc} Weight (α)	Lr	Lr Decay	Number of Iterations	Image Size	Negative Ratio	L_{loc} Weight (α)	Lr	Lr Decay	Number of Iterations
ICDAR 2011	384	3	0.2	5×10^{-3}	1k,2k	3k	768	6	1.0	5×10^{-4}	2k	3.5k
ICDAR 2013	384	3	0.2	5×10^{-3}	0.5k, 1k	1.4k	768	6	1.0	5×10^{-4}	1k, 2k	3k
ICDAR 2015	384	3	0.2	5×10^{-3}	2k, 4k	6.5k	768	6	1.0	5×10^{-4}	5k, 10k	15k
ReCTS 2019	384	3	0.2	5×10^{-3}	40k, 80k	120k	768	6	1.0	5×10^{-4}	100k, 200k	220k
MLT 2019	384	3	0.2	5×10^{-3}	21k, 42k	63k	768	6	1.0	5×10^{-4}	50k, 100k, 150k	170k
MSRA-TD500	384	3	0.2	5×10^{-3}	2.5k	4k	768	6	1.0	5×10^{-4}	3k, 6k	7k

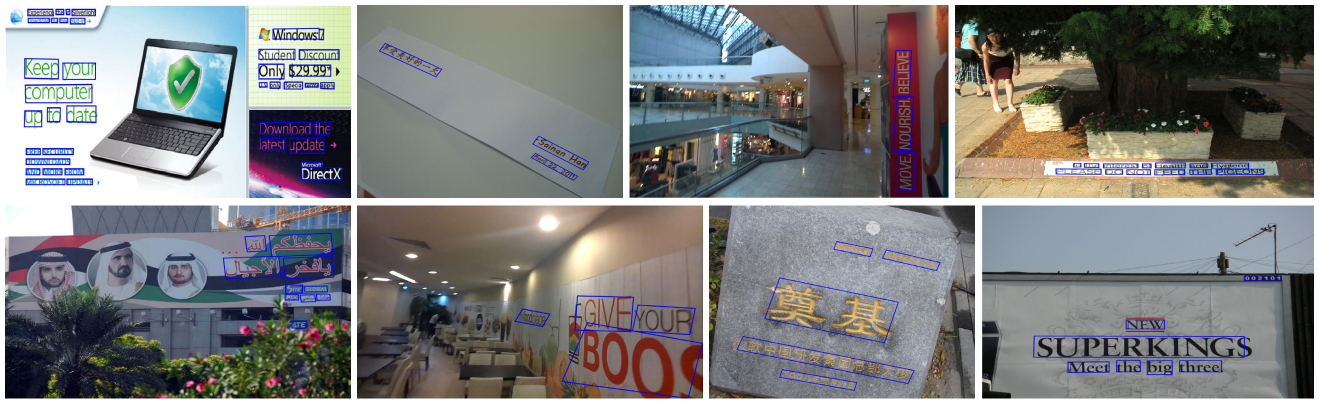
[1] $\mathbf{Lr_{decay}}$ points to the iterations in which the learning rate is decayed.

TABLE 5. Smartphone specifications.

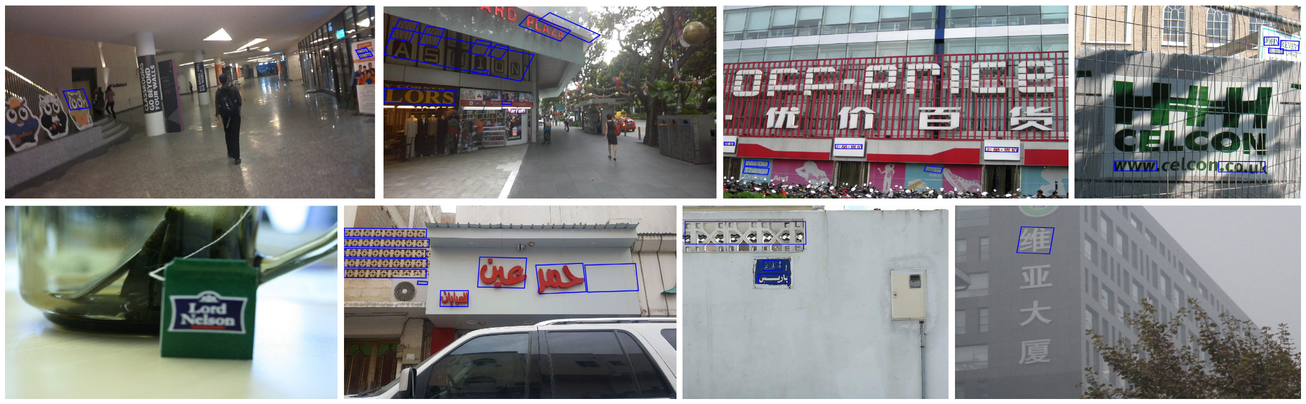
	Smartphones		
	Motorola Moto G6 Play	Asus Zenfone 5	Xiaomi Mi 9T
Platform	Android 9	Android 9	Android 10
CPU architecture	ARMv7	ARMv7	x86_64
Memory	3GB	4GB	6GB
Processors	Octa-core 8x1.4GHz Cortex-A53	Octa-core 4x1.8GHz Kryo 260 Gold 4x1.6GHz Kryo 260 Silver	Octa-core 2x2.2GHz Kryo 470 Gold 6x1.8GHz Kryo 470 Silver

(88.79MB) are approaches with models relative smaller than most of the methods. They obtained F-measures of just 54.00 and 69.60, which are 27.72 and 12.32 percentage points less than Pelee-Text++.

Pelee-Text++ outperformed Pelee-Text on ICDAR 2015, and reached competitive results against state-of-the-art approaches (FOTS [34], PMTD [32] and PDR [27]); our method was 6.3× and 4.99× smaller than PMTD and FOTS, respectively. Figure 4b shows the effectiveness of each method vs their model size, our proposal is a promising approach considering the trade-off between efficacy and model size. Pelee-Text++ was very close to the best methods



(a) Successful cases.



(b) Failure cases.

FIGURE 3. Detection results of our proposed method: (a) correct detections; (b) some failure cases.

regarding to the Precision; nevertheless, it still missed words as the Recall score shows. Our model had problems to detect vertical word cases and when it does, very low confidence values are assigned (see Figure 3b)).

Finally, as we can observe in Table 7 and Table 8, almost all of the methods are based on VGG-16 [46] and ResNet [15] networks, which lead to heavy models in terms of disk usage. Furthermore, some of them used these networks along with a Feature Pyramid approach [30], and even creating several branches to improve text detection without concerns on their model size hampering their use on devices with computational constraints.

On the other hand, Pelee-Text++ is a promising tiny neural network architecture that reached competitive results using a light architecture favorable for mobile devices, as Figure 3a shows. Additionally, our single scale versions of 768 and 1024 achieved good results and they ran at 23.25 and 15.06 FPS, respectively, while our multi-scale version (384, 768, 1024 and 1536) ran at 3.65 FPS.

B. DETECTING MULTI-ORIENTED MULTI-LINGUAL TEXT

We have evaluated Pelee-Text++ on ReCTS [64], MSRA-TD500 [59] and MLT 2019 [40], which are more challenging scenarios than ICDAR 2013 and ICDAR 2015 datasets. ReCTS and MSRA-TD500 are datasets containing English and Chinese text, and they are line-level datasets, i.e., their bounding boxes cover lines of text instead of single words. On the other hand, MLT 2019 is a dataset containing images with texts belonging to ten different languages: Chinese, Japanese, Korean, English, French, Arabic, Italian, German, Bangla, and Hindi (Devanagari). This multi-lingual task also takes into account punctuation and math symbols present in several images.

Table 9 shows the effectiveness of diverse methods on MSRA-TD500. Pelee-Text++ achieved very promising results and placed very close to DRRG [65], which is the best approach over this dataset. The Precision obtained for

TABLE 6. Text detection results on ICDAR 2011 dataset.

Methods	**P (%)**	**R (%)**	**F1 (%)**
Text-CNN [17]	91.00	74.00	82.00
SynthText [12]	94.30	76.90	84.70
TextBoxes [28]	88.00	82.00	85.00
MobText [7]	97.40	94.81	**96.09**
Pelee-Text_MS	94.99	87.26	90.96
Pelee-Text++_MS	94.33	88.27	91.20

TABLE 7. Text detection results on ICDAR 2013 dataset.

Methods	Backbone	ICDAR 2013 P(%)	R(%)	F1(%)	FPS	Framework
TextBoxes++ [29]	VGG-16	74.00	86.00	80.00	11.60	Caffe
FOTS [34]	ResNet-50 + FPN	——	——	88.23	23.90	Caffe
BorderSemantics [56]	DenseNet	91.50	87.10	89.20	——	——
*MaskTextSpotter [38]	ResNet-50 + FPN	95.00	88.60	91.70	4.60	Pytorch
ATRR [51]	SE-VGG16	93.70	89.70	91.70	10.00	Caffe
GISCA [4]	VGG-16 + UNet	92.80	91.40	92.10	——	——
BATDN [68]	VGG-16 + FPN	94.44	90.16	92.25	——	——
*FTPN [31]	ResNet-101 + FPN	93.20	91.90	92.50	——	Tensorflow
PMTD [32]	ResNet-50 + FPN	——	——	93.40	——	Pytorch
*CRAFT [1]	VGG-16	97.40	93.10	**95.20**	8.60	Pytorch
FCN [67]	VGG-16	88.00	78.00	83.00	$<$ 1.00	Caffe
MobText [7]	MobileNetv2	88.38	66.67	76.00	1.82	Tensorflow
MobileNetv2+UNet [9]	MobileNetv2 + UNet	83.00	72.00	76.00	20.00	——
Pelee-Text_768	PeleeNet	80.13	79.87	80.00	18.64	Caffe
Pelee-Text++_768	PeleeNet	87.04	73.53	79.72	23.25	Caffe
CornerLoc_MS [39]	VGG-16 + FPN	92.00	84.40	88.00	1.00	Pytorch
TextBoxes++_MS [29]	VGG-16	91.00	84.00	88.00	2.30	Caffe
*PixelLink_2s_MS [8]	VGG-16	88.60	87.50	88.10	——	Tensorflow
FOTS_MS [34]	ResNet-50 + FPN	——	——	92.50	——	Caffe
Pelee-Text_MS	PeleeNet	88.41	82.28	85.24	2.93	Caffe
Pelee-Text++_MS	PeleeNet	92.42	80.04	85.78	3.65	Caffe

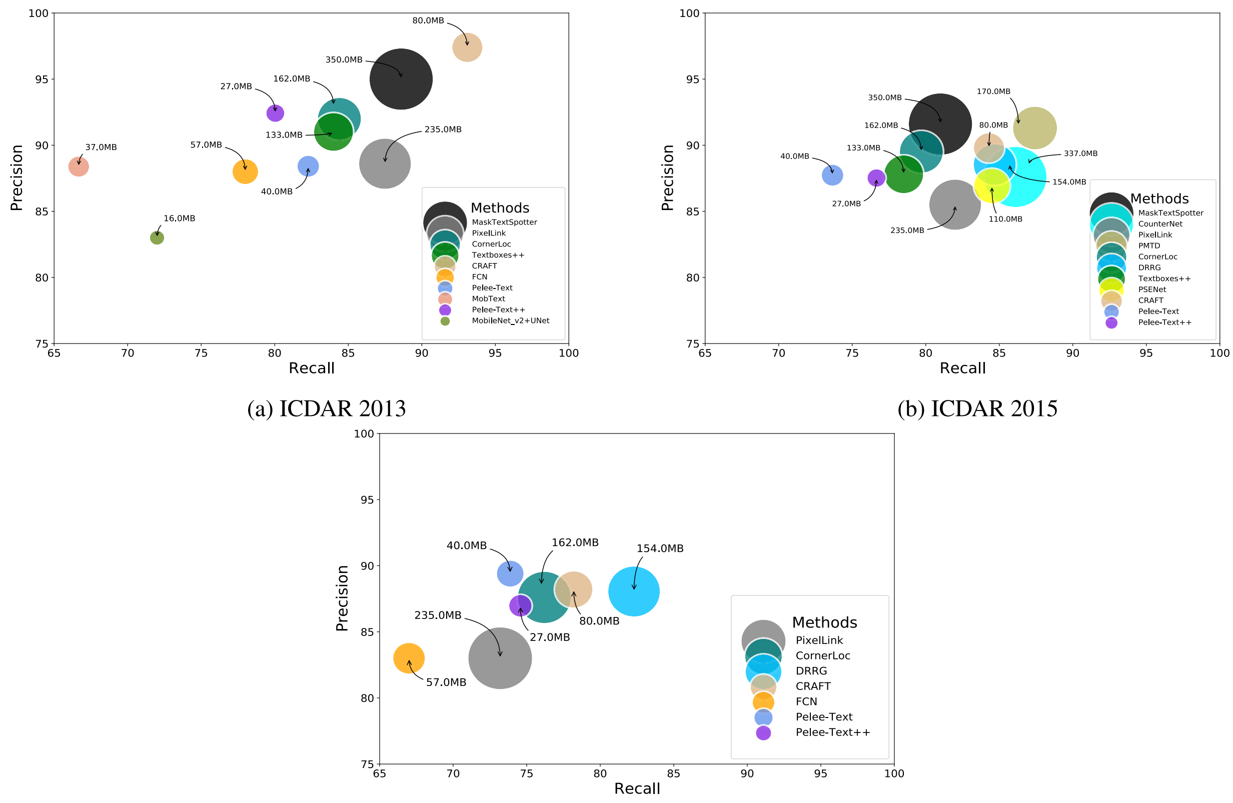
[1] "——" means unreported values on their papers.
[2] "*" methods used DetEval evaluation protocol.

TABLE 8. Text detection results on ICDAR 2015 dataset.

Methods	Backbone	ICDAR 2015 P(%)	R(%)	F1(%)	FPS	Framework
FTPN [31]	ResNet-101 + FPN	68.20	78.00	72.80	3.33	Tensorflow
TextBoxes++ [29]	VGG-16	76.70	87.20	81.70	11.60	Caffe
MSR [57]	ResNet-50	86.60	78.40	82.30	4.3	Tensorflow
TextSnake [36]	VGG-16 + FPN	84.90	80.40	82.60	1.10	Tensorflow
GA-DAN [62]	PVANet	85.60	81.60	83.50	——	——
PixelLink_2s [8]	VGG-16	85.50	82.00	83.70	3.00	Tensorflow
PSENet [50]	ResNet-50	86.92	84.50	85.69	1.60	Pytorch
MaskTextSpotter [38]	ResNet-50 + FPN	91.60	81.00	86.00	4.80	Pytorch
DRRG [65]	VGG-16 + FPN	88.53	84.69	86.56	——	Pytorch
CRAFT [1]	VGG-16	89.80	84.30	86.90	8.60	Pytorch
CountourNet [52]	ResNet-50 + FPN	87.60	86.10	86.90	3.50	Pytorch
LOMO [63]	ResNet-50 + FPN	91.30	83.50	87.20	——	——
BATDN [68]	VGG-16 + FPN	89.63	84.93	87.22	——	——
GISCA [4]	VGG-16 + UNet	89.10	85.50	87.30	——	——
ATRR [51]	SE-VGG16	89.20	86.00	87.60	10.00	Caffe
FOTS [34]	ResNet-50 + FPN	91.00	85.17	87.99	7.80	Caffe
PMTD [32]	ResNet-50 + FPN	91.30	87.43	89.33	——	Pytorch
PDR [27]	ResNet-50 + FPN	83.73	96.13	89.51	3.7	——
FCN [67]	VGG-16	71.00	43.00	54.00	$<$ 1.00	Caffe
OctMLT [37]	ResNet-Blocks + Shuffle	——	——	69.40	——	——
Pelee-Text_1024	PeleeNet	85.19	72.27	78.20	11.67	Caffe
Pelee-Text++_1024	PeleeNet	81.62	78.29	79.92	15.06	Caffe
TextBoxes++_MS [29]	VGG-16	87.80	78.50	82.90	2.30	Caffe
TextField_MS [55]	VGG-16	83.90	84.30	84.10	1.80	Caffe
CornerLoc_MS [39]	VGG-16 + FPN	89.50	79.70	84.30	1.00	Pytorch
MaskTTD [35]	ResNet-50 + FPN	86.60	87.60	87.10	1.60	——
LOMO_MS [63]	ResNet-50 + FPN	87.80	87.60	87.70	——	——
FOTS_MS [34]	ResNet-50 + FPN	91.85	87.92	**89.84**	——	Caffe
Pelee-Text_MS	PeleeNet	87.73	73.66	80.08	2.93	Caffe
Pelee-Text++_MS	PeleeNet	87.52	76.65	81.72	3.65	Caffe

[1] "——" means unreported values on their papers.

Pelee-Text++ was comparable with DRRG, but recall needs to be improved. This is a dataset with several vertical text cases, scenario in which our method has problems to detect and/or assign high confidence values. Nonetheless, as we can see in Figure 4c in terms of disk usage, the model of DRRG is 5.7× larger than Pelee-Text++.



(a) ICDAR 2013

(b) ICDAR 2015

(c) MSRA-TD500

FIGURE 4. **Effectiveness vs model size.**

TABLE 9. **Text detection results on MSRA-TD500.**

Methods	**P (%)**	**R (%)**	**F1 (%)**
FCN [67]	83.00	67.00	74.00
GA-DAN [62]	80.50	71.10	75.50
PixelLink_2s [8]	83.00	73.20	77.80
TextSnake [36]	83.20	73.90	78.30
BorderSemantics [56]	83.00	77.40	80.10
TextField [55]	87.40	75.90	81.30
GISCA [4]	86.30	77.10	81.40
CornerLoc [39]	87.60	76.20	81.50
MSR [57]	87.40	76.70	81.70
CRAFT [1]	88.20	78.20	82.90
MaskTDD [35]	85.70	81.10	83.30
ATRR [51]	85.20	82.10	83.60
DRRG [65]	88.05	82.30	**85.08**
Pelee-Text_MS	89.40	73.88	80.90
Pelee-Text++_MS	86.97	74.57	80.30

TABLE 10. **Text detection results on MLT 2019.**

Methods	**P (%)**	**R (%)**	**F1 (%)**
LOMO [63]	87.75	79.80	**83.59**
PMTD [32]	87.47	78.12	82.53
CASCADE-RCNN [3]	82.26	74.85	78.38
CRAFT [1]	81.42	62.73	70.86
PSENet [50]	73.52	59.59	65.83
PeleeText++_MS	78.65	57.63	66.51

[1] Results taken from the competition dashboard (https://rrc.cvc.uab.es/?ch=15&com=evaluation&task=1 – As of July 2020).

On the ReCTS [64] dataset, our method obtained a F-measure of 82.53%. However, there not exist works in the literature providing results over this dataset. Then, we do not have information to compare our proposal against other approaches in terms of effectiveness and model size.

Our proposal reached a very promising trade-off between efficacy and model size. We can notice that it was able to detect both English and Chinese texts, as depicted in Figure 3. Experimental results showed that Pelee-Text++ outperformed well-known methods, such as PixelLink [8] and TextSnake [36] not only in effectiveness but also in terms of the size of the model. Concerning to the best methods and their models size, CRAFT [1] and DRRG [65] obtained better results than Pelee-Text++; nevertheless, their models were 2.96 and 5.7 times larger than our proposal.

Nowadays, MLT 2019 [40] is one of the most challenging datasets for multilingual text detection; but, there not exist much papers showing the performance of methods over this dataset. For this reason, Table 10 shows results taken from the competition dashboard considering only the methods with an associated paper. As we can see, LOMO [63] is the state-of-the-art method with a F-measure of 83.59%, followed by PMTD [32], CASCADE-RCNN [3], CRAFT [1], and PSENet [50] with 82.53%, 78.38%, 70.86%, and 65.83%

TABLE 11. Processing time on smartphones.

Smartphone	Method	Time by Scale (seconds) 300	384	768	1024
MOTOROLA MOTO-G6 PLAY	MobText	**0.77±0.072**	−−	−−	−−
	Pelee-Text++	0.99 ± 0.026	**1.54±0.048**	**5.76±0.188**	**10.24±0.308**
	Pelee-Text	1.40 ± 0.030	2.25 ± 0.080	8.21 ± 0.230	Memory Problems
	CRAFT	7.37 ± 0.050	12.15 ± 0.060	50.45 ± 1.199	Memory Problems
	PSENet	5.08 ± 0.034	8.32 ± 0.077	42.70 ± 0.356	Memory Problems
	TextBoxes++	10.16 ± 0.078	12.33 ± 0.478	Memory Problems	Memory Problems
ASUS ZENFONE 5	MobText	**0.22±0.034**	−−	−−	−−
	Pelee-Text++	0.35 ± 0.029	**0.91±0.072**	**2.99±0.176**	**3.59±0.125**
	Pelee-Text	0.51 ± 0.039	1.32 ± 0.057	4.57 ± 0.416	8.65 ± 1.126
	CRAFT	2.08 ± 0.022	3.28 ± 0.092	13.09 ± 0.234	24.93 ± 0.310
	PSENet	1.69 ± 0.033	2.55 ± 0.177	14.25 ± 0.705	34.27 ± 1.119
	TextBoxes++	3.60 ± 0.231	6.83 ± 0.789	Memory Problems	Memory Problems
XIAOMI MI 9T	MobText	**0.14±0.024**	−−	−−	−−
	Pelee-Text++	0.34 ± 0.022	**0.50±0.036**	**1.74±0.075**	**3.00±0.270**
	Pelee-Text	0.51 ± 0.033	0.76 ± 0.089	2.66 ± 0.212	4.41 ± 0.440
	CRAFT	1.87 ± 0.045	2.69 ± 0.043	8.86 ± 0.144	15.22 ± 0.181
	PSENet	1.85 ± 0.024	2.61 ± 0.031	9.42 ± 0.182	17.32 ± 0.466
	TextBoxes++	2.82 ± 0.185	4.29 ± 0.089	Memory Problems	Memory Problems

"−−" MobText was designed for running only with 300×300 images.

of F-measure, respectively. It is worth mentioning that CASCADE-RCNN [3] was cited in the competition site as the associated paper for one of the results. The submission, however, does not have a well-explained description about the modifications of this network for detecting multilingual text.

Furthermore, with respect to the remaining approaches, as mentioned before, all those methods are based on deep architectures, such as VGG and ResNet. For instance, LOMO (without information about its model size) and PMTD (170MB) use ResNet-50 along with FPN, whereas CRAFT (80MB) and PSENet (110MB) use VGG-16.

Additionally, LOMO and PMTD, the top-2 methods on this dataset, use several branches for specific tasks in order to improve results but impacting directly on their model size. In contrast, Pelee-Text++, with a model size of 27MB, obtained better results than PSENet (4.07× larger), and it was close to CRAFT (2.96× larger). Similarly as we described before, our network has to be improved with regard to the detection of vertical align text. One possible research venue concerns the use of a different training protocol specifically tailored to this multi-lingual scenario.

C. RESULTS ON MOBILE DEVICES

This section presents the results of our proposal against five methods on three smartphones (Motorola Moto-G6, Asus Zenfone 5 and Xiamoi Mi T9). Details about the smartphones and the app are described in Section IV-C.

For comparison purposes, we used the models available on the authors' official GitHub. Along with our proposal, the methods considered for evaluation were: Pelee-text [6], MobText [7], TextBoxes++,[5] CRAFT,[6] and PSENet.[7] These methods were implemented using different frameworks: Pelee-Text, Pelee-Text++, and Textboxes++ used Caffe, MobText was implemented on Tensorflow, whereas CRAFT and PSENet used Pytorch. These experiments only considered the processing time of each model to generate its outputs without taking into consideration neither pre or post processing procedures of each approach.

Table 11 presents the processing time and standard deviation of each model over different smartphones to process an image using four different scales: 300 × 300, 384 × 384, 768 × 768, and 1024 × 1024. As we can see, PeleeText++ was the only method that ran on all three smartphones without causing memory issues with the four scales. Moreover, it had the fastest inference time in all scenarios, except for an image of size 300 × 300, where the best was MobText. As we explained in previous sections, MobText is a network specifically designed for working with image input size of 300×300. Additionally, this method has limitations for scenarios with oriented text where quadrilateral representation is needed.

TextBoxes++, the method with the larger model size (133MB) in this experiment, was not able to run with image size of 768 and 1024 in any of the three smartphones, causing memory problems. On Motorola Moto-G6 with 3GB of memory RAM, Pelee-Text, CRAFT, and PSENET caused memory issues with an image size of 1024. Compared to CRAFT, one of the state-of-the-art methods on the datasets evaluated in this work, Pelee-Text++ was 7.44×, 7.89×, and 8.76× faster using an input size of 300, 384, and 768, respectively.

On Asus Zenfone 5 with 4GB of memory RAM, all the methods executed faster. Over this environment Pelee-Text, CRAFT, and PSENet could perform inference even with an image size of 1024. Respecting to the inference time on Moto-G6, Pelee-Text++ had a time reduction of 77.77% on the 300 scale image, 41% on the 384, 48.09% on the 768 and 65% with an image size of 1024. Furthermore, considering

[5]https://github.com/MhLiao/TextBoxes_plusplus (As of July 2020).
[6]https://github.com/clovaai/CRAFT-pytorch (As of July 2020).
[7]https://github.com/whai362/PSENet (As of July 2020).

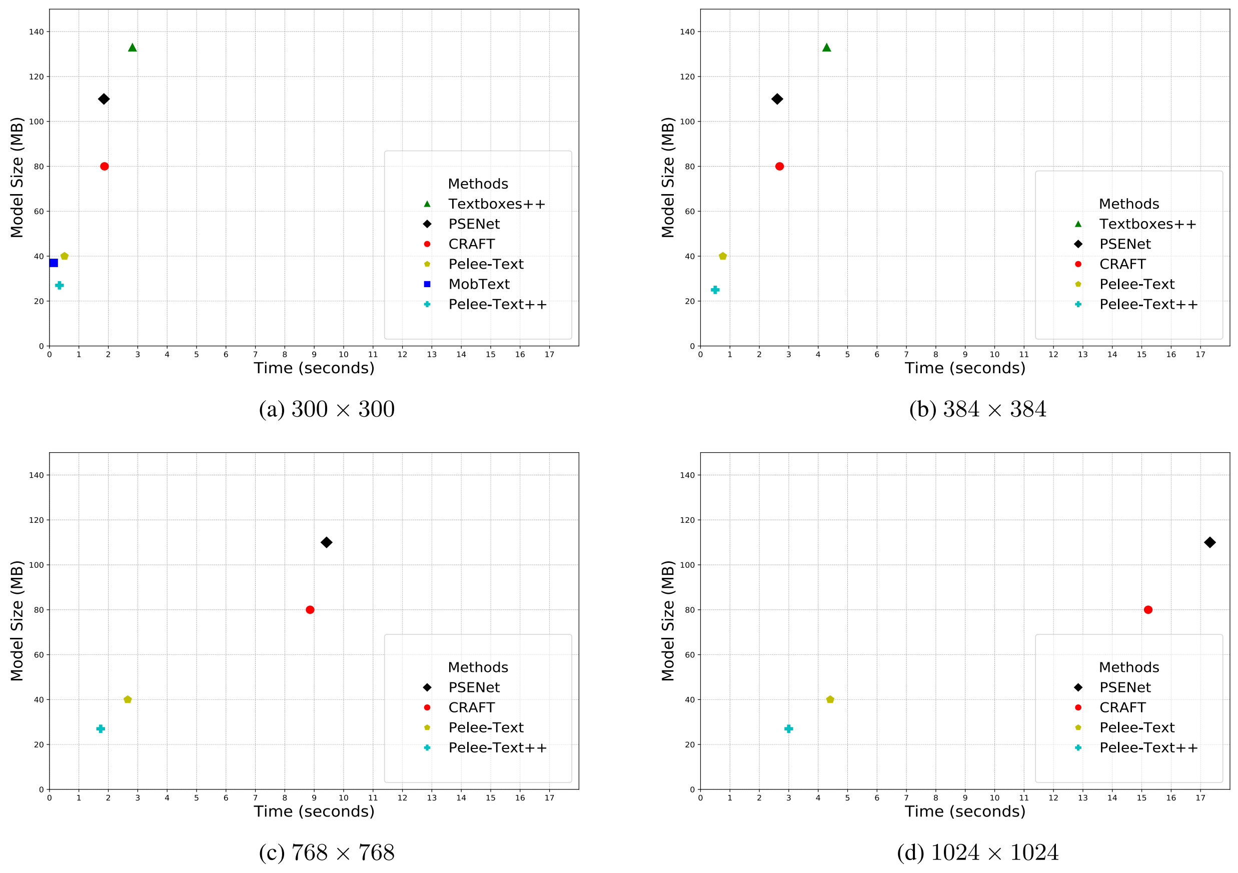


FIGURE 5. Efficiency on XIAOMI Mi T9 using different scales for input image.

the best time of each method, our proposal was 1.46×, 5.94×, 4.83×, 10.29× faster than Pelee-Text, CRAFT, PSENet, and TextBoxes++, respectively. In contrast, considering the worst time of each one, which is when an input image of 1024 is used, Pelee-Text++ was just 41.50% of the processing time of Pelee-Text, 14.40% of the CRAFT's inference time, and 10.48% of the PSENet.

Figure 5 shows the performance of the assessed methods on XIAOMI Mi T9, which has 6GB of memory RAM. Even on this smartphone, TextBoxes++ could not complete its inference when the input image was 768 and 1024 because of memory issues. On the other hand, MobText reached the best time when an input image of 300, but it was tested only over this scenario due to its nature of being specifically designed for this input size.

Additionally, we can see how the gap between tiny vs large models increases with a bigger input image. Without considering MobText and TextBoxes++, because of the previous related issues, the gap between Pelee-Text++ (best time) vs the worst time was of 2.48 seconds for an image of size 300, while for 1024 there exist a huge difference of 14.32 seconds.

It is difficult to compare methods implemented with different frameworks and programming languages, but the experiments performed on three commercial smartphones showed a big gap in efficiency between the approaches based on large deep learning models and our proposal. Finally, based on these results and the effectiveness obtained by the methods in the evaluated datasets, Pelee-Text++ seems to be a very promising architecture for text detection being a light, fast, and competitive approach vs state-of-the-art methods. Pelee-text++ with a model size of 27MB was capable of processing 2.94 FPS in a real mobile setup, being at least 5.5 times faster than CRAFT, which is one of the best methods in several datasets.

VI. CONCLUSION

Unlike other works in the text detection field, which has been increasing the model size of their approaches adopting the use of very deep architectures or even fusing several task-specific branches, we presented a very promising lightweight neural network for dealing with scene text detection and with special focus on measuring the performance of our proposal against state-of-the-art methods in mobile devices.

We proposed Pelee-Text++, a mobile-based convolutional neural network specifically designed for detecting text which usually has a longer aspect ratio. One of the main problems is to capture text with some types of orientation that are not fully covered by rectangular bounding boxes. For that reason, our approach uses quadrilaterals instead.

The experimental results over five well-known datasets involving born-digital images, horizontal/vertical text aligned, and multi-lingual setups in real scene images, show a great performance of the Pelee-Tex++ in terms of effectiveness and efficiency. Our network obtained competitive results against state-of-the-art methods and showed a very promising trade-off between effectiveness and model size. On GPU, it runs at 23.25, 15.06 and 3.65 FPS for our 768, 1024 and multi-scale versions, respectively. More important, our proposal demonstrated its efficiency in three smartphones, being faster than well-known methods. Pelee-Text++ has a model size of just 27 Megabytes and processes 2.94 FPS on a smartphone being at least 5.5 times faster than CRAFT, which is one of the best methods in the text detection area.

Concisely, we demonstrated that our approach is more appropriate for restrictive computing scenarios. Pelee-Text++ presents outperforming results in terms of time processing and model size reduction, becoming a promising approach to avoid disk usage and RAM memory consumption problems in mobile devices. Despite of the competitive results reached by Pelee-Text++ in arbitrary-oriented (horizontal and vertical text) and multi-oriented text; one of the limitations of our proposal is that Pelee-Text++ is not capable of working well on datasets presenting irregular and curved text because of its nature, i.e., prediction of quadrilaterals that do not fit well in these types of scenarios. We intend to address these issues in future works.

Additionally, future research efforts will focus on smart data augmentation strategies, for instance, rotation, cropping, among others, for detecting the missing cases, such as text in images with the presence of blurring and occlusion, detecting (near)-vertical textual elements. Furthermore, the proposal of training and testing protocols for multi-lingual scenarios, and evaluation of the impact of model compression methods for obtaining a more compact neural network without loss of effectiveness.

REFERENCES

[1] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, "Character region awareness for text detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9365–9374.

[2] A. Furkan Biten, R. Tito, A. Mafla, L. Gomez, M. Rusiñol, M. Mathew, C. V. Jawahar, E. Valveny, and D. Karatzas, "ICDAR 2019 competition on scene text visual question answering," *CoRR*, vol. abs/1907.00490, pp. 1–8, Sep. 2019. [Online]. Available: http://arxiv.org/abs/1907.00490

[3] Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving into high quality object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6154–6162.

[4] M. Cao, Y. Zou, D. Yang, and C. Liu, "Gisca: Gradient-inductive segmentation network with contextual attention for scene text detection," *IEEE Access*, vol. 7, pp. 62805–62816, 2019.

[5] T. Choudhary, V. Mishra, A. Goswami, and J. Sarangapani, "A comprehensive survey on model compression and acceleration," *Artif. Intell. Rev.*, vol. 53, no. 7, pp. 5113–5155, Oct. 2020.

[6] M. A. Córdova, L. G. L. Decker, J. L. Flores-Campana, A. A. dos Santos, and J. S. Conceição, "Pelee-text: A tiny convolutional neural network for multi-oriented scene text detection," in *Proc. 18th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2019, pp. 400–405.

[7] L. Decker, A. Pinto, J. Campana, M. Neira, A. Santos, J. Conceição, M. Angeloni, L. Li, and R. Torres, "MobText: A compact method for scene text localization," in *Proc. 15th Int. Joint Conf. Comput. Vis., Imag. Comput. Graph. Theory Appl.*, G. M. Farinella, P. Radeva, and J. Braz, Eds., vol. 5, Feb. 2020, pp. 343–350.

[8] D. Deng, H. Liu, X. Li, and D. Cai, "Pixellink: Detecting scene text via instance segmentation," in *Proc. AAAI Conf. Art. Intell.*, 2018, pp. 6773–6780.

[9] K. Fu, L. Sun, X. Kang, and F. Ren, "Text detection for natural scene based on mobilenet V2 and u-net," in *Proc. IEEE Int. Conf. Mechatronics Autom. (ICMA)*, Aug. 2019, pp. 1560–1564.

[10] Á. González, L. M. Bergasa, and J. J. Yebes, "Text detection and recognition on traffic panels from street-level imagery using visual appearance," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 1, pp. 228–238, Feb. 2014.

[11] J. Greenhalgh and M. Mirmehdi, "Recognizing text-based traffic signs," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 3, pp. 1360–1369, Jun. 2015.

[12] A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic data for text localisation in natural images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2315–2324.

[13] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," in *Proc. 4th Int. Conf. Learn. Representations (ICLR)*, San Juan, Puerto Rico, May 2016, pp. 1–14.

[14] D. He, X. Yang, C. Liang, Z. Zhou, A. G. Ororbia, D. Kifer, and C. L. Giles, "Multi-scale FCN with cascaded instance aware segmentation for arbitrary oriented word spotting in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 474–483.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[16] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2961–2969.

[17] T. He, W. Huang, Y. Qiao, and J. Yao, "Text-attentional convolutional neural network for scene text detection," *IEEE Trans. Image Process.*, vol. 25, no. 6, pp. 2529–2541, Jun. 2016.

[18] W. He, X.-Y. Zhang, F. Yin, and C.-L. Liu, "Multi-oriented and multi-lingual scene text detection with direct regression," *IEEE Trans. Image Process.*, vol. 27, no. 11, pp. 5406–5419, Nov. 2018.

[19] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*. [Online]. Available: http://arxiv.org/abs/1704.04861

[20] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 2261–2269.

[21] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy, "Speed/accuracy trade-offs for modern convolutional object detectors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 3296–3297.

[22] R. Huang, J. Pedoeem, and C. Chen, "YOLO-LITE: A real-time object detection algorithm optimized for non-GPU computers," in *Proc. IEEE Int. Conf. Big Data*, Dec. 2018, pp. 2503–2510.

[23] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5mb model size," 2016, *arXiv:1602.07360*. [Online]. Available: https://arxiv.org/abs/1602.07360

[24] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, L. G. I. Bigorda, S. R. Mestre, J. Mas, D. F. Mota, J. A. Almazan, and L. P. de las Heras, "ICDAR 2013 robust reading competition," in *Proc. 12th Int. Conf. Document Anal. Recognit.*, Aug. 2013, pp. 1484–1493.

[25] D. Karatzas, L. Gomez-Bigorda, A. Nicolaou, S. Ghosh, A. Bagdanov, M. Iwamura, J. Matas, L. Neumann, V. R. Chandrasekhar, S. Lu, F. Shafait, S. Uchida, and E. Valveny, "ICDAR 2015 competition on robust reading," in *Proc. Int. Conf. Document Anal. Recognit.*, 2015, pp. 1156–1160.

[26] E. Li, L. Zeng, Z. Zhou, and X. Chen, "Edge AI: On-demand accelerating deep neural network inference via edge computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 447–457, Jan. 2020.

[27] J. Li, Z. Zhou, Z. Su, S. Huang, and L. Jin, "A new parallel detection-recognition approach for end-to-end scene text extraction," in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sep. 2019, pp. 1358–1365.

[28] M. Liao, B. Shi, X. Bai, X. Wang, and W. Liu, "TextBoxes: A fast text detector with a single deep neural network," in *Proc. AAAI Conf. Art. Intell.*, 2017, pp. 4161–4167.

[29] M. Liao, B. Shi, and X. Bai, "TextBoxes++: A single-shot oriented scene text detector," *IEEE Trans. Image Process.*, vol. 27, no. 8, pp. 3676–3690, Aug. 2018.

[30] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 936–944.

[31] F. Liu, C. Chen, D. Gu, and J. Zheng, "FTPN: Scene text detection with feature pyramid based text proposal network," *IEEE Access*, vol. 7, pp. 44219–44228, 2019.

[32] J. Liu, X. Liu, J. Sheng, D. Liang, X. Li, and Q. Liu, "Pyramid mask text detector," *CoRR*, vol. abs/1903.11800, pp. 1–10, Mar. 2019. [Online]. Available: http://arxiv.org/abs/1903.11800

[33] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Proc. ECCV*, 2016, pp. 21–37.

[34] X. Liu, D. Liang, S. Yan, D. Chen, Y. Qiao, and J. Yan, "FOTS: Fast oriented text spotting with a unified network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5676–5685.

[35] Y. Liu, L. Jin, and C. Fang, "Arbitrarily shaped scene text detection with a mask tightness text detector," *IEEE Trans. Image Process.*, vol. 29, pp. 2918–2930, 2020.

[36] S. Long, J. Ruan, W. Zhang, X. He, W. Wu, and C. Yao, "TextSnake: A flexible representation for detecting text of arbitrary shapes," in *Proc. 15th Eur. Conf. Comput. Vis.*, 2018, pp. 19–35.

[37] A. Lundgren, D. Castro, E. Lima, and B. Bezerra, "OctShuffleMLT: A compact octave based neural network for end-to-end multilingual text detection and recognition," in *Proc. Int. Conf. Document Anal. Recognit. Workshops (ICDARW)*, vol. 4, Sep. 2019, pp. 37–42.

[38] P. Lyu, M. Liao, C. Yao, W. Wu, and X. Bai, "Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes," in *Proc. 15th Eur. Conf. Comput. Vis.*, 2018, pp. 71–88.

[39] P. Lyu, C. Yao, W. Wu, S. Yan, and X. Bai, "Multi-oriented scene text detection via corner localization and region segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7553–7563.

[40] N. Nayef, C.-L. Liu, J.-M. Ogier, Y. Patel, M. Busta, P. N. Chowdhury, D. Karatzas, W. Khlif, J. Matas, U. Pal, and J.-C. Burie, "ICDAR2019 robust reading challenge on multi-lingual scene text detection and recognition—RRC-MLT-2019," in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sep. 2019, pp. 1582–1587.

[41] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6517–6525.

[42] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: https://arxiv.org/abs/1804.02767

[43] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention—MICCAI*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham, Switzerland: Springer, 2015, pp. 234–241.

[44] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

[45] A. Shahab, F. Shafait, and A. Dengel, "ICDAR 2011 robust reading competition challenge 2: Reading text in scene images," in *Proc. Int. Conf. Document Anal. Recognit.*, Sep. 2011, pp. 1491–1496.

[46] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Representations*, 2015, pp. 1–14.

[47] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proc. AAAI*, 2017, pp. 1–12.

[48] S. Tian, S. Lu, and C. Li, "Wetext: Scene text detection under weak supervision," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1492–1500.

[49] R. J. Wang, X. Li, and C. X. Ling, "Pelee: A real-time object detection system on mobile devices," in *Proc. Adv. Neural Inf. Process. Syst.*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2018, pp. 1963–1972.

[50] W. Wang, E. Xie, X. Li, W. Hou, T. Lu, G. Yu, and S. Shao, "Shape robust text detection with progressive scale expansion network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9336–9345.

[51] X. Wang, Y. Jiang, Z. Luo, C.-L. Liu, H. Choi, and S. Kim, "Arbitrary shape scene text detection with adaptive text region representation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 6449–6458.

[52] Y. Wang, H. Xie, Z.-J. Zha, M. Xing, Z. Fu, and Y. Zhang, "ContourNet: Taking a further step toward accurate arbitrary-shaped scene text detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11753–11762.

[53] S. Wen, "Translation analysis of English address image recognition based on image recognition," *EURASIP J. Image Video Process.*, vol. 2019, no. 1, p. 11, Dec. 2019.

[54] B. Wu, A. Wan, F. Iandola, P. H. Jin, and K. Keutzer, "SqueezeDet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 446–454.

[55] Y. Xu, Y. Wang, W. Zhou, Y. Wang, Z. Yang, and X. Bai, "TextField: Learning a deep direction field for irregular scene text detection," *IEEE Trans. Image Process.*, vol. 28, no. 11, pp. 5566–5579, Nov. 2019.

[56] C. Xue, S. Lu, and F. Zhan, "Accurate scene text detection through border semantics awareness and bootstrapping," in *Proc. 15th Eur. Conf. Comput. Vis. (ECCV)*, in Lecture Notes in Computer Science, vol. 11220, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Munich, Germany: Springer, Sep. 2018, pp. 370–387.

[57] C. Xue, S. Lu, and W. Zhang, "MSR: Multi-scale shape regression for scene text detection," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 989–995.

[58] H. Yang and C. Meinel, "Content based lecture video retrieval using speech and video text information," *IEEE Trans. Learn. Technol.*, vol. 7, no. 2, pp. 142–154, Apr. 2014.

[59] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu, "Detecting texts of arbitrary orientations in natural images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1083–1090.

[60] C. Yi and Y. Tian, "Scene text recognition in mobile applications by character descriptor and structure configuration," *IEEE Trans. Image Process.*, vol. 23, no. 7, pp. 2972–2982, Jul. 2014.

[61] C. Yi, Y. Tian, and A. Arditi, "Portable camera-based assistive text and product label reading from hand-held objects for blind persons," *IEEE/ASME Trans. Mechatronics*, vol. 19, no. 3, pp. 808–817, Jun. 2014.

[62] F. Zhan, C. Xue, and S. Lu, "GA-DAN: Geometry-aware domain adaptation network for scene text detection and recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9105–9115.

[63] C. Zhang, B. Liang, Z. Huang, M. En, J. Han, E. Ding, and X. Ding, "Look more than once: An accurate detector for text of arbitrary shapes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10552–10561.

[64] R. Zhang, M. Yang, X. Bai, B. Shi, D. Karatzas, S. Lu, C. V. Jawahar, Y. Zhou, Q. Jiang, Q. Song, N. Li, K. Zhou, L. Wang, D. Wang, and M. Liao, "ICDAR 2019 robust reading challenge on reading chinese text on signboard," in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sydney, NSW, Australia, Sep. 2019, pp. 1577–1581.

[65] S.-X. Zhang, X. Zhu, J.-B. Hou, C. Liu, C. Yang, H. Wang, and X.-C. Yin, "Deep relational reasoning graph network for arbitrary shape text detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 9699–9708.

[66] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6848–6856.

[67] Z. Zhang, C. Zhang, W. Shen, C. Yao, W. Liu, and X. Bai, "Multi-oriented text detection with fully convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 4159–4167.

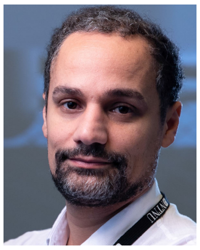
[68] Z. Zhong, L. Sun, and Q. Huo, "A teacher-student learning based born-again training approach to improving scene text detection accuracy," in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sep. 2019, pp. 281–286.

[69] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8697–8710.

MANUEL CÓRDOVA received the B.Sc. degree in systems engineer from the National University of Loja (UNL), Ecuador, in 2010, and the M.Sc. degree in computer science from Unicamp, in 2015. He is currently pursuing the Ph.D. degree with the University of Campinas (UNICAMP), Brazil. He worked as a Professor with the Department of Systems Engineering, National University of Loja (UNL), Ecuador, from 2016 to 2018. He is a member of the RECOD Lab, where he has been working in research projects involving machine learning and computer vision.

ALLAN PINTO (Member, IEEE) received the B.Sc. degree in computer science from the University of São Paulo (USP), Brazil, in 2011, the M.Sc. and Ph.D. degrees in computer science from the University of Campinas (Unicamp), Brazil, in 2013 and 2018, respectively. A part of his Ph.D. was accomplished at the University of Notre Dame, Notre Dame, IN, USA, in which he worked on different topics such as Presentation Attack Detection in Biometric Systems, Content-based Image Retrieval, and Multimedia Forensics. He is currently a Postdoctoral Researcher with the University of Campinas (Unicamp), Brazil. His research interests include image and video analysis, pattern recognition, machine learning, computer vision, multimedia forensics, biometrics, content-based aimage retrieval, remote sensing, sports sciences, and data sciences. He is a member of the Editorial Board of the *Forensic Science International: Reports*.

HELIO PEDRINI (Senior Member, IEEE) received the B.Sc. degree in computer science and the M.Sc. degree in electrical engineering from the University of Campinas, Brazil, and the Ph.D. degree in electrical and computer engineering from Rensselaer Polytechnic Institute, Troy, NY, USA. He is currently a Professor with the Institute of Computing, University of Campinas. His research interests include image processing, computer vision, pattern recognition, machine learning, computational geometry, geometric modeling, computer graphics, and scientific visualization. He is a member of the Institute of Electrical and Electronics Engineers (IEEE) and of the Brazilian Computer Society (SBC).



RICARDO DA SILVA TORRES (Member, IEEE) received the B.Sc. degree in computer engineering and the Ph.D. degree in computer science from the University of Campinas, Brazil, in 2000 and 2004, respectively. He has been serving as the Coordinator of Master Program in Simulation and Visualization from NTNU since August 2020. He has been developing multidisciplinary eScience research projects involving Multimedia Analysis, Multimedia Retrieval, Machine Learning, Databases, Information Visualization, and Digital Libraries. He used to hold a position as a Professor at the University of Campinas, Brazil, from 2005 to 2019. He is currently a Professor in visual computing with the Norwegian University of Science and Technology (NTNU). He is author or coauthor of more than 200 articles in refereed journals and conferences and serves as a PC Member for several international and national conferences. He has been serving as a Senior Associate Editor for the IEEE SIGNAL PROCESSING LETTERS and an Associate Editor for the *Pattern Recognition Letters*.