IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 8 May 2023, accepted 28 May 2023, date of publication 7 June 2023, date of current version 13 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3283931

RESEARCH ARTICLE

Resource-Aware Scene Text Recognition Using Learned Features, Quantization, and Contour-Based Character Extraction

OLUTOSIN AJIBOLA ADEMOLA[1], EDUARD PETLENKOV[2], (Member, IEEE), AND MAIRO LEIER[1]

[1]Embedded AI Research Laboratory, Department of Computer Systems, Tallinn University of Technology, 19086 Tallinn, Estonia
[2]Centre for Intelligent Systems, Department of Computer Systems, Tallinn University of Technology, 19086 Tallinn, Estonia

Corresponding author: Olutosin Ajibola Ademola (olutosin.ademola@taltech.ee)

This work was supported in part by the "Information and Communications Technology (ICT) Program" through the European Union via the European Social Fund, in part by the Innovation Fund Denmark under the Project "Green Roll-on/Roll-off (RORO) Shipping through Digital Innovation (ROROGREEN)" under Grant 0177-00022B, and in part by the Estonian Research Council under Grant PRG658.

ABSTRACT Scene texts serve as valuable information for humans and autonomous systems to make informed decisions. Processing scene texts poses significant difficulties for computer systems due to several factors, primarily due to variations in image characteristics. These factors make it very challenging for computer systems to accurately detect and interpret scene texts, despite being easily understandable to humans. To address this problem, scene text detection and recognition methods leverage computer vision and/or deep learning methods. Deep learning methods require substantial resources, including computing power, memory, and energy. As such, their use in real-time embedded applications, particularly those that run on integer-only hardware, is very challenging due to the resource-intensive nature of these methods. In this paper, we developed an approach to address this challenge and to showcase its effectiveness, we trained end-to-end models for shipping container number detection and recognition. By doing so, we were able to demonstrate the accuracy and reliability of our proposed method for processing scene texts on integer-only hardware. Our efforts to optimize the models yielded impressive results. We reduced the model size by a factor of 3.8x without significantly affecting the models' performance. Moreover, the optimized models were 1.6x faster, and the maximum RAM usage was 6.6x lower than the base models. These results demonstrate the efficiency and practicality of our approach for scene text processing on integer-only embedded hardware.

INDEX TERMS Deep learning model quantization, integer-only hardware, resource-constrained devices, scene text detection, scene text recognition.

I. INTRODUCTION

We carried out a thorough review of journal search and indexing databases to examine current state-of-the-art methods for scene text detection and recognition. Based on our analysis, we found that no prior work has been done to address the challenges of implementing these methods on integer-only embedded hardware. This highlights the significance and novelty of this research work.

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

The emergence of resource-efficient hardware for deep learning applications, which only supports integer-based operations and operates under stringent storage, memory, and computational power constraints, has been a significant development.

The possibility of accurately detecting and recognizing text in natural scene images has created endless use cases in different embedded applications. One predominant area is autonomous systems. Autonomous systems have a wide range of applications and one of the most prominent areas is their use in various tasks that require intelligent decision-making capabilities. These tasks may involve

intelligent navigation, traffic management, parcel sorting, ticketing, natural language translation, and guiding systems, among others.

Scene text detection and recognition techniques are based on computer vision and/or deep learning methods, and deep learning methods are resource intensive in terms of computing power, memory, and energy usage. Consequently, implementing these methods in real-time embedded applications, particularly those that operate on integer-only hardware, can be highly challenging due to these resource requirements. Different methods have been proposed for text detection in natural scenes [1], [2], [3].

Classic methods (i.e., computer vision-based techniques) utilize sliding windows or connected component analysis to detect the region of text [4], [5], [6], [7]. The sliding window uses a window of multiple scales that moves through the receptive field of the image. The receptive regions (i.e., the text region candidates) are cropped and a machine learning classifier such as Support Vector [8], Random Forest [9], or AdaBoost [10], etc., is trained to predict the text candidates.

Connected component analysis utilizes manual filters to extract salient features such as edges, text texture, boundary points, and text color, among others, from images. These features are used to train a machine learning model [11], [12], [13], [14].

Due to the rise in the adoption of deep learning technology influenced by improved computing resources, availability of big data, etc., unparalleled results have been achieved in almost all computer vision-related tasks that require artificial intelligence such as scene text detection, text recognition, image classification, multi-object detection, etc [15].

Deep learning methods outperform computer vision-based methods because distinctive features are automatically learned using kernel filters instead of relying on manually designed filters to extract fundamental features. As the tasks become more complex, such as in the case of scene text where there are variations in light intensity, surface roughness, low-quality images, etc., the effectiveness of hand-crafted filters tends to decrease. This is because these filters are not able to handle the intricacies of such complex tasks, thus, leading to reduced efficiency.

Several deep learning-based algorithms have been proposed for detecting scene text [16], [17], [18], [19], [20]. These methods rely on state-of-the-art region-based convolutional neural network frameworks for object detection. The region proposal network is responsible for computing the objectness score of the region containing the text region using sets of predetermined anchors. Proposed regions, also known as anchors, are cropped and then fed into the fully connected layer to predict the location of the text region.

Other deep learning methods proposed involve the use of state-of-the-art image segmentation algorithms that classify the text using pixels such that the pixels of the regions containing text are classified as the text class and vice-versa [21], [22], [23], [24], [25].

The high computational and memory requirements of these methods make them expensive, which limits their use in embedded applications running on integer-only hardware. Our proposed method for scene text detection and recognition involves using learned features, a quantization technique with offset, and contour-based character extraction. Our method is designed to be resource-aware, making it suitable for use in integer-only hardware where resources such as memory and compute power are limited.

In summary, our main contributions are as follows:

- We introduced an 8-bit quantization technique for text detection and recognition models. This makes it possible to deploy the models on embedded hardware that only supports integer operations, without a notable drop in performance.
- We introduced a quantization bias to the ground-truth labels to offset the quantization-induced error and improve the accuracy of the models.
- We introduced a module specifically for text orientation detection to improve our recognition pipeline's ability to process text that is oriented both vertically and horizontally.

This paper is divided into several sections, each focusing on different aspects of scene text detection and recognition. The first section provides an introduction, which includes a discussion of existing methods and their limitations, as well as the potential use cases for autonomous systems. Additionally, this section highlights our novel contributions to addressing the challenges of deploying these methods on integer-only hardware.

In section two, we describe the problems associated with implementing text detection and recognition models on integer-only embedded hardware. We also explain the novelty of our work and the need for resource-efficient solutions.

Section three provides a comprehensive review of the state-of-the-art methods for scene-text detection and recognition, highlighting the limitations of each approach. In section four, we present our proposed method in detail, which addresses the challenges of deploying scene text detection and recognition models on integer-only embedded hardware.

Section five discusses the dataset used in our experiments, its source, and the development hardware we used. In section six, we present the results of our experiments in detail. Finally, section seven provides a concluding discussion on the need for resource-aware text detection and recognition, the effectiveness of our proposed method, and a summary of the results achieved.

II. PROBLEM STATEMENT

There are numerous potential applications for scene text detection and recognition in real-time embedded systems. In this section, we will showcase a case study to illustrate this point. In Fig. 1, there are different trucks carrying shipping containers. The containers have unique identification numbers, known as cargo identification numbers, which consist of both numbers and letters.

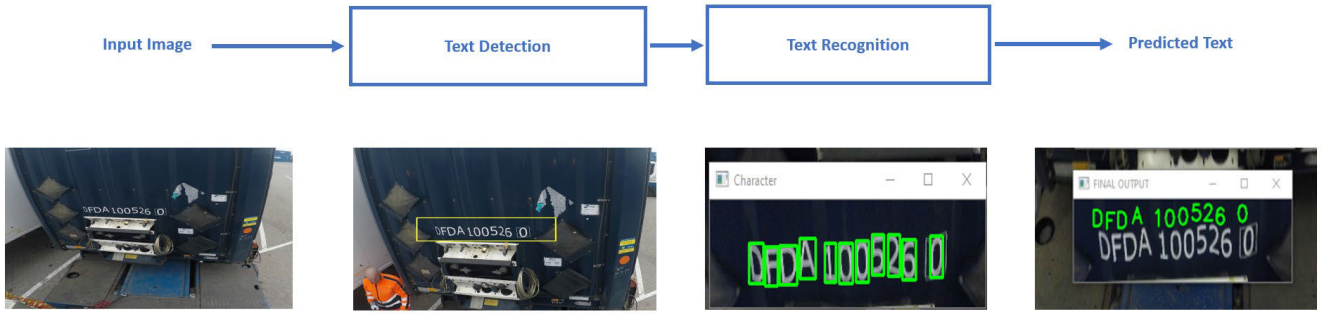


FIGURE 1. Text detection and recognition stages involved in textual information extraction in natural scene images.

Our goal is to efficiently and accurately track every container being transported from the port terminal to the decks of the ship. This ensures that each container, regardless of its size or type, is placed on the designated deck. This objective arises from the need for proper stowage management, which is critical for ensuring the safety of the crew and the successful delivery of the transported containers containing valuable goods.

Identifying containers by their unique cargo identification number presents a challenge in text detection and recognition. While we have reviewed existing methods, none of them meet the specific needs of our use case. Our requirements are particularly strict, as we need a solution that is compatible with integer-only hardware and efficient in terms of storage, computational power, and memory usage.

III. STATE OF THE ART

Scene text recognition methods rely on text detection algorithms. As such, the accuracy depends on how accurately the region of interest is estimated. In this section, we will discuss the state-of-the-art methods for scene text detection and recognition.

As discussed in the introduction section, scene text detection and recognition methods are based on two techniques —computer vision [4], [5], [6] and deep learning [17], [19], [25]. Deep learning methods have been proven to outperform computer vision-based approaches [26], [27], [28], therefore, our work focuses on deep learning-based techniques.

A. TEXT DETECTION

Jaderberg et al. [16] proposed a single pipeline for text detection and recognition. The detection module in their approach relies on a region proposal network. Another method, DeepText [17], utilizes a unified framework that combines a convolutional neural network for region proposal and detection. The region proposal component in DeepText employs an inception module. In [18], the authors used Faster R-CNN for detecting multi-orientation text. Faster R-CNN also incorporates a region proposal network.

Tian et al. [19] introduced CTPN (Connectionist Text Proposal Network), a text proposal network that combines convolutional neural network (CNN) and recurrent neural network (RNN) with an anchor mechanism for fixed-width proposals. Zhang et al. [21] combined a Fully Connected Network (FCN) with text line hypotheses to detect multi-oriented text. In [22], scene text detection was approached as a segmentation problem, utilizing holistic and multi-channel prediction.

TextEdge [24] implemented a multi-oriented FCN scene text detector that employs region segmentation and edge classification. Zhou et al. [25] introduced EAST, an Efficient and Accurate Scene Text detector, which utilizes a fully convolutional network for scene text detection. Rong et al. [29] proposed a dense text localization network combined with context reasoning for scene text retrieval.

B. TEXT RECOGNITION

Jaderberg et al. [16] introduced deep convolutional neural networks for word-level recognition. Their approach differs from our work, which employs a character-based classifier for scene text recognition. In [26], an end-to-end text spotting method was proposed, utilizing a convolutional recurrent neural network. This unified pipeline requires both ground truth labels for the scene text and bounding box labels.

Bagi et al. [30] introduced a lightweight text spotter that utilizes a lightweight deep neural network for word-level recognition. Cao et al. [31] employed a fully convolutional neural network with an attention module for detecting small text. In [29], the authors utilized a recurrent neural network for the recognition module.

Liu et al. [32] introduced an adaptive bezier-curve network for end-to-end text spotting. The text spotter was further quantized with different bit widths to enhance the network's inference time. However, the emphasis was not placed on the model size and peak runtime memory of the model.

Previous studies have shown that an end-to-end scene text detection and recognition system can employ a single pipeline for both tasks [33], [34], [35]. However, to create a resource-efficient text detection and recognition model suitable for hardware limited to integer operations, certain requirements need to be fulfilled.

Firstly, the model should be lightweight, typically ranging from a few kilobytes to megabytes in size. Secondly, it should have a small memory footprint, typically a few kilobytes to megabytes, to ensure compatibility with the device's capacity.

Finally, the model must be optimized to exclusively support integer-based operations, aligning with the hardware's limitations.

The existing state-of-the-art methods are not well-suited for implementation on integer-only hardware, such as Edge TPUs or microcontrollers. In order to address this challenge, we propose a deep learning-based method that is specifically tailored for such hardware. Our approach takes advantage of learned features, utilizes a quantization technique with offset, and integrates contour-based character extraction.

By being resource-aware, our method is specifically designed to be suitable for integer-only hardware, where limitations in resources such as memory and compute power are prevalent. This resource awareness allows our method to optimize the utilization of available resources, making efficient use of the limited memory and computational capabilities of integer-only hardware. Thus, our method offers a viable solution for enabling effective text detection and recognition on integer-only embedded hardware.

IV. PROPOSED METHOD

A. OVERALL ARCHITECTURE

Scene text recognition methods typically follow a two-stage approach, consisting of text detection and recognition stages, as depicted in Fig. 1. During the text detection stage, the system localizes the region of the text in the image by determining the bounding box coordinates. This stage is of utmost importance as the subsequent recognition stage heavily relies on accurate text detection.

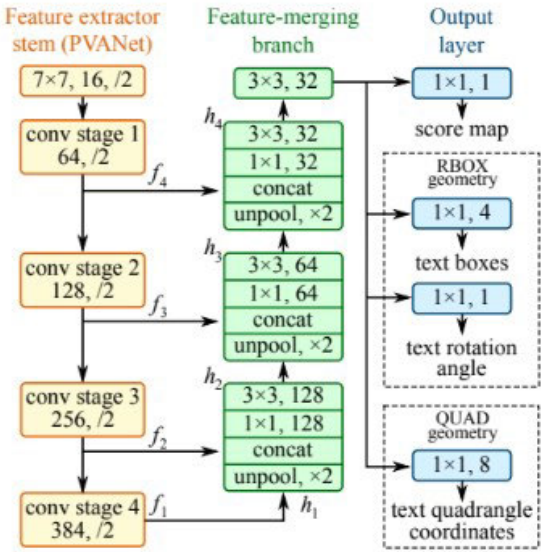


FIGURE 2. The original EAST architecture [25].

B. TEXT DETECTION

Our text detection method is architecturally inspired by the EAST (Efficient and Accurate Scene Text) model [25]. EAST, known as the Efficient and Accurate Scene Text Detector, utilizes a fully convolutional neural network to predict the region of interest where text is present. EAST lacks a recognition module. We selected the EAST architecture due to its excellent suitability for our specific use case. Moreover, the EAST architecture seamlessly integrates into our pipeline, as illustrated in Fig. 1.

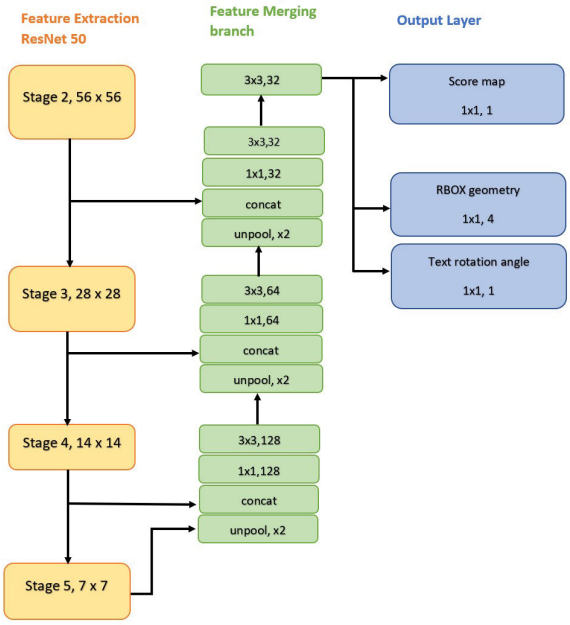


FIGURE 3. The modified EAST architecture using ResNet50 [36] as the base network for the extraction of features.

Several factors influence the suitability and effectiveness of scene text detectors in different applications, and the characteristics and type of the scene text are particularly influential. The architecture consists of three stages: feature extraction, feature merging, and output generation, as illustrated in Fig. 2.

In our modified architecture, we opted for ResNet-50 [36] as the base network for feature extraction, deviating from the original EAST architecture that employed PVANET [37], as depicted in Fig. 3. We made this selection for the following reasons:

- It is faster because it uses a 1 × 1 kernel filter in its bottleneck design. This design reduces the number of matrix multiplication and network parameters, therefore, reducing the time it takes during propagation.
- ResNet-50 uses a global average pooling rather than fully connected layers. Thus, reduces the size of the model.
- ResNet-50 generalized well on our dataset compared to VGG16 and VGG19.

ResNet-50 is composed of 50 layers, which are divided into five stages of convolution blocks. Fig. 4 illustrates this architecture. The first stage contains a convolution block with 64 filters of size 7 × 7 and a stride of 2, as well as a max pooling layer with a stride of 2. The input image size is "320 px × 320 px." The second stage comprises three sets of three convolution blocks. These blocks consist of 64 filters of size 1 × 1, 64 filters of size 3 × 3, and 512 filters of size

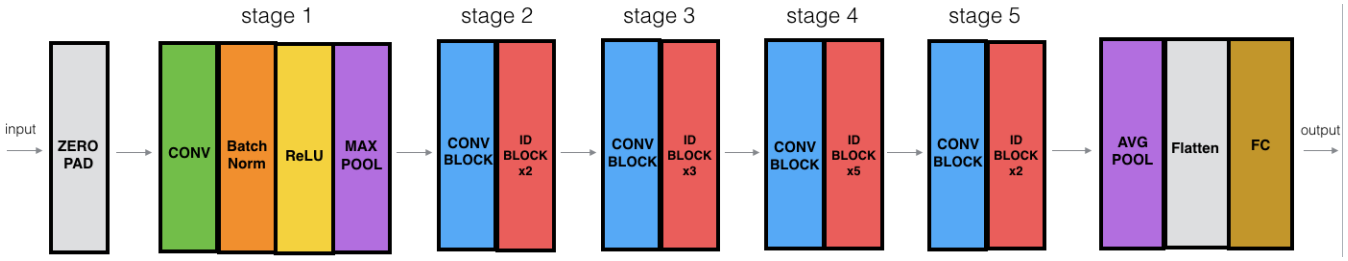


FIGURE 4. The architecture of ResNet50 used as the base network [36].

1×1. The third stage contains four sets of three convolution blocks.

These blocks consist of 128 filters of size 1×1, 128 filters of size 3×3, and 512 filters of size 1×1. The fourth stage consists of six stacks of three convolution blocks. These blocks consist of 256 filters of size 1×1, 256 filters of size 3×3, and 1024 filters of size 1×1. The fifth stage consists of three stacks of three convolution blocks. These blocks consist of 512 filters of size 1×1, 256 filters of size 3×3, and 2048 filters of size 1×1. The feature merging stage uses the intermediate output of each ResNet-50 stage to reduce the computational complexity of processing all merged features at once, as shown in Fig. 3.

The output of each stage is upsampled so that the output size (i.e., the feature map size) will be of the same size as the input of the stage for concatenation along the channel of the feature maps. 1×1 and 3×3 kernel filters are applied. This is repeated for the other stages. A 3×3 kernel filter is applied to the output of the last upsampled stage which serves as the input of the output stage. The output stage consists of a series of 1×1 kernel filters to produce the confidence score and the coordinates of the region of interest of the text, as shown in Fig. 3.

Our approach focuses primarily on obtaining two key features: the confidence score of text presence, represented by the score map, and the coordinates of the corresponding bounding boxes. These bounding boxes can correspond to horizontal or vertical text regions, as depicted in Fig. 5.

Accurately determining the bounding box type is a crucial aspect of our method. We achieve this by utilizing the bounding boxes generated at the output stage. The precise estimation of the bounding box type plays a pivotal role in the subsequent text recognition stage. It assists in correctly identifying the first and last characters of a word, which is essential for the reconstruction of the words.

C. QUANTIZATION

The parameters of the text detection model are typically represented using 32-bit full-precision floating point values. However, when it comes to quantization for integer-only hardware, text detectors can be highly sensitive to dynamic quantization, where only the model weights are integers, and even more sensitive to full integer quantization, where all parameters, including weights, biases, and activations, are

FIGURE 5. The horizontal and vertical text orientations.

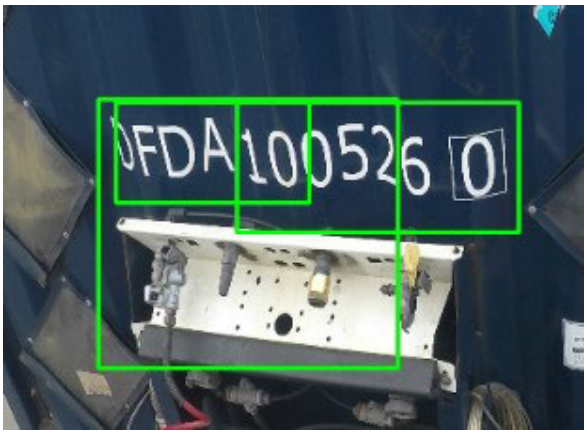


FIGURE 6. Multiple bounding boxes overlapping the scene text caused by quantization-induced error.

integers. To address this challenge, we introduced a quantization offset during the generation of ground-truth labels.

The purpose of the tolerance is to account for the error introduced by quantization, as illustrated in Fig. 6. To ensure compatibility with integer-only hardware, we applied quantization to the text detection model using an 8-bit symmetric signed integer quantizer.

The quantizer takes a 32-bit input float tensor X_f (e.g., the weight matrix of the model), and each parameter is quantized to an 8-bit signed integer using both "equation (1)," and "equation (2)."

$$m_f = \frac{\frac{2^7-1}{2}}{\max(|X_f|)}, \tag{1}$$

$$q_{8bit} = round(m_f X_f). \tag{2}$$

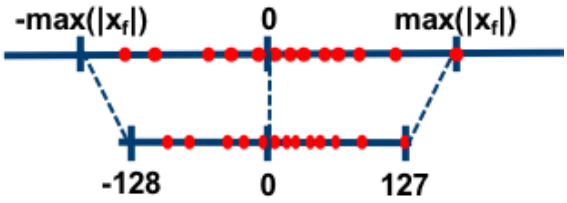


FIGURE 7. The mapping of floating point weight values to 8-bit quantized signed integer representations.

where m_f is the scaling factor and q_{8it} is the quantized output whose range is limited by the absolute value of X_f as shown in Fig. 7.

D. TEXT RECOGNITION

The text recognition stage relies on the output of the text detection model as described in Fig. 1. The text recognition pipeline has two phases (the preprocessing and recognition phases) as shown in Fig. 8.

FIGURE 8. The two stages involved in text recognition.

The preprocessing phase determines the type of bounding box (i.e., the text orientation) using the text coordinates produced by the detection model. This phase extracts the region of interest from the image and removes unwanted contours.

The extraction method is based on a contour-based extraction algorithm that we developed. This algorithm computes the contour of each character, discards unwanted contours, and uses the computed contours to extract the characters.

The characters that have been extracted from the detected text region are then inputted into the text recognition model. The architecture of our lightweight text recognition model incorporates convolutional and dense blocks, which are outlined in detail in Table 1. After the individual characters are predicted, they are aggregated and combined to form the complete recognized text, as shown in Fig. 1.

To ensure compatibility with integer-only hardware, the text recognition model undergoes full quantization using the 8-bit symmetric signed integer quantizer. This quantization process is defined by ‘‘equation (1)’’ and ‘‘equation (2)’’.

V. EXPERIMENTS

In our experiment, we utilized a dataset comprising 2000 images. Out of these, 1500 images were allocated for training the text detection model, while the remaining 500 images were reserved for testing purposes. It is important to note that the images used in this experiment are proprietary and specifically developed for this project.

The input images for the text detection model were standardized to a size of 320 × 320 pixels. The dataset consists of various images of containers, each displaying their unique cargo identification number, as depicted in Fig. 1.

For the purpose of training our text recognition model, we extracted a total of 8750 images. These images were resized to a dimension of 64 pixels by 64 pixels. Each image contained one of the 35 uppercase characters, including numbers (0-9) and letters (A-Z) excluding ‘O’. There were precisely 250 images per character, resulting in a well-balanced dataset.

TABLE 1. Architecture of the text recognition model.

Layer Type	Output Size	Parameters
CONV2D	(None, 64, 64, 32)	864
BN	(None, 64, 64, 32)	96
MAXPOOL2D	(None, 32, 32, 32)	0
CONV2D	(None, 32, 32, 64)	18432
BN	(None, 32, 32, 64)	192
MAXPOOL2D	(None, 16, 16, 64)	0
CONV2D	(None, 16, 16, 64)	36928
BN	(None, 16, 16, 64)	192
FLATTEN	(None, 16384)	0
DENSE	(None, 64)	1048576
BN	(None, 64)	192
DENSE1	(None, 35)	2240
ACTIVATION	(None, 35)	0
	TOTAL PARAMETERS	1107712

Out of the extracted images, we allocated 7000 for training the text recognition model, while the remaining 1750 images were set aside for testing purposes.

To ensure compatibility with our desired integer-only model, we selected the Google Coral Development Board as the target hardware. This board is equipped with Quad Cortex-A53 and Cortex-M4F processors, along with an Edge TPU coprocessor. Additionally, it provides 1 GB of RAM and 8 GB of flash memory [38].

Our text detection and recognition models were trained until no further improvements in performance were observed. Nevertheless, we are unable to deploy these models on the target hardware due to its support for only integer-based operations, as well as the strict requirements of our application, which include a small model size footprint, fast inference, high accuracy, efficient peak RAM usage, and computational efficiency. Therefore, further optimization is necessary to meet these requirements.

Quantization plays a significant role in the performance of text detection and recognition models. It refers to the process of reducing the precision of numerical values in a model, typically from floating-point to integer representations. However, quantization can introduce errors and affect the accuracy of the models.

To overcome this challenge, we introduced a quantization offset to the ground-truth labels. This offset is designed to compensate for the errors induced by quantization, ensuring that the model’s predictions align closely with the original floating-point values.

By incorporating the quantization offset, we aim to minimize the impact of quantization on the performance of our text detection and recognition models. This approach allows us to achieve a balance between model optimization for integer-only hardware and preserving the accuracy and reliability of the model’s predictions.

We applied quantization to both the text detection and recognition models, reducing the precision of the model's parameters such as weights, biases, and activations. Specifically, the parameters were converted from their original 32-bit floating-point representation to 8-bit signed integer representations.

By quantizing the models, we aimed to make them compatible with integer-only hardware and improve their efficiency in terms of memory usage and computational cost. Quantization helps to reduce the model size and allows for faster inference, making it suitable for resource-constrained environments such as edge TPUs or microcontrollers.

The performance of the quantized models was evaluated by measuring their accuracy and overall effectiveness using five key evaluation metrics as described in Table 2 and Table 3. These metrics include the model performance, peak RAM footprint, model size, computational cost, and inference time.

The evaluation considered the performance metrics outlined in Table 2 and Table 3, allowing us to analyze and compare the impact of quantization on various aspects of the model's performance.

By examining these metrics, we gained deep insights into the trade-offs and improvements achieved through the quantization process, enabling us to make informed decisions regarding the suitability of the models for resource-constrained hardware.

TABLE 2. Performance evaluation metrics for validating quantized model applicability.

	Text Detection		Text Recognition	
Metrics	Original	**Quantised**	Original	**Quantised**
Model Size (MB)	96.21	**24.83**	0.88	**0.23**
Inference Time (ms)	2356.00	**1450.77**	3.59	**2.18**
Computational Cost (MFLOP)	15072.50	**0**	20.20	**0**
Peak RAM Usage (MB)	286.23	**40.63**	5.04	**3.29**

TABLE 3. Performance evaluation metrics for validating quantized model applicability.

	Text Detection		Text Recognition	
	Mean Loss		Accuracy	
Metric	Original	**Quantised**	Original	**Quantised**
Model Performance (%)	25.51	**26.23**	99.73	**99.62**

VI. RESULT AND DISCUSSION

Recognizing text in natural scenes is a challenging task due to several factors, including variations in image quality, diverse device types, varying lighting conditions, different text orientations, and the presence of clustered text in scene images. The accurate prediction of text heavily relies on the performance of text detection methods.

It is crucial to highlight that the effectiveness of text detection algorithms significantly impacts the accuracy and precision of text recognition methods. Therefore, ensuring high-quality text detection is essential for achieving reliable and robust text recognition results.

To assess the suitability of the quantized models for our intended purpose, we conducted a comprehensive evaluation that considered various key performance metrics. These metrics are essential in determining the applicability of the models on the target hardware.

The suitability of the quantized models was evaluated by measuring their accuracy and overall effectiveness in text detection and recognition tasks. Additionally, we assessed the peak RAM usage, which indicates the maximum amount of memory consumed by the models during operation. Model size, another important metric, reflects the storage requirements of the models.

Furthermore, we analyzed the computational cost associated with running the quantized models, considering factors such as the number of operations performed and the processing power required. Lastly, we measured the inference time, which indicates the speed at which the models can process input data and provide output.

By evaluating these performance metrics, we gained valuable insights into the practicality and efficiency of the quantized models for deployment on resource-constrained devices, especially integer-only hardware. This information is crucial for designing effective and optimized solutions that meet the requirements of our target hardware.

To conduct a comprehensive comparison between the base models and their quantized counterparts, we utilized the key performance indicators presented in Table 2 and Table 3. These indicators were derived from a series of experiments conducted using diverse sample data, ensuring a representative evaluation.

The results presented in Table 2 and Table 3 are derived from a thorough evaluation conducted through multiple experiments using diverse sample data. This rigorous approach of averaging the performance metrics over various experiments enhances the reliability and validity of the reported findings.

By using different data samples, we obtain a comprehensive evaluation that provides a more accurate representation of the models' performance. This ensures that the conclusions drawn from the comparison between the base models and their quantized counterparts are robust and applicable in real-time embedded applications.

The model size refers to the amount of flash memory required to store the model's parameters, such as weights and biases. By default, the weights are stored using a 32-bit full-precision float. In our approach, we applied an 8-bit symmetric quantizer, as described in Figure 7 and Equations (1) and (2), to both the text detection and recognition models. As a result, we achieved a 3.87x reduction in the flash size required to store the quantized text detection model.

Similarly, the quantized text recognition model demonstrated a 3.82x reduction in model size compared to the uncompressed text recognition model, as indicated in Table 2. Notably, the quantized models maintained their performance, as evidenced by the results presented in Table 3.

We evaluated the text detection model's performance using the mean loss metric, which is a combination of the dice and intersection over union (IoU) losses. The lower the mean loss value, the better the model's performance.

FIGURE 9. End-to-end text detection and recognition results of our proposed method.

The quantized text detection model demonstrated a 2% increase in mean loss compared to the base model. On the other hand, the quantized text recognition model showed no significant decrease (only a 0.11% decrease) in performance despite having undergone significant model compression.

The speed of a model during inference is affected by multiple factors, including but not limited to the number of reads and write operations, memory bit width, and types of operations performed. We achieved an improvement of approximately 1.65x in model speed for both quantized models.

In real-time embedded applications, the availability of random access memory (RAM) is crucial for the application's smooth operation without interruptions or delays. RAM is used to store dynamic data that the application requires to function properly.

Deep learning models, such as our base text detection and recognition models, are computationally expensive, especially in terms of RAM resource usage. As indicated in Table 2, the text detection model requires at least 286.23 MB of RAM, while the text recognition model requires at least 5.04 MB. This results in a total RAM requirement of 291.27 MB for the end-to-end pipeline.

Our proposed method enabled us to achieve a significant reduction in RAM usage for the quantized models, resulting in a total of only 43.92 MB of RAM required. This represents a compression factor of 6.63x when compared to the RAM requirements of the base models.

We need to acknowledge a limitation of our proposed method, which is its applicability to less clustered text in scene images. This limitation arises from the need to introduce a quantization bias when preparing the ground-truth labels to compensate for the quantization-induced error.

It's important to note that scene text can vary greatly, and our method may not be suitable for all types of scene text.

VII. CONCLUSION

The increasing utilization of deep learning technology in computer vision tasks owes to a multitude of factors, including advancements in computing power, the availability of vast datasets, and the development of sophisticated algorithms.

Deep learning technology has brought about remarkable breakthroughs, especially in the domain of scene text detection and recognition. The process involves the precise localization of text regions within scene images and subsequent identification of the text contained within these regions.

Scene text detection and recognition have become pronounced due to the rise in the number of portable and embedded devices. These devices are capable of running different intelligent applications. Some of these applications require understanding textual information in scene images for decision-making. Such applications include an intelligent transportation system, text-to-speech, auto navigation, object detection, etc.

The emergence of resource-efficient hardware for deep learning applications, that only supports integer-based operations and operates under stringent constraints on storage, memory, and computational power, has been a significant development.

The current state-of-the-art methods for scene text detection and recognition rely heavily on deep learning approaches that demand significant resources, such as computing power, memory, and energy. As such, the implementation of these methods in real-time embedded applications, especially those

operating on integer-only hardware, poses a considerable challenge.

We developed a resource-efficient method to tackle this issue. To demonstrate its effectiveness and suitability for integer-only hardware, we trained end-to-end models specifically designed for detecting and recognizing shipping containers. Subsequently, these models were deployed on the target hardware.

We demonstrated the accuracy and reliability of our proposed method for processing scene texts on this piece of hardware. Our efforts to optimize the models yielded impressive results as shown in Table 2 and Table 3.

Our optimization efforts resulted in a significant reduction in model size, achieving a compression factor of 3.8x while maintaining comparable performance to the base models. Additionally, the optimized models exhibited a 1.6x increase in speed, accompanied by a substantial decrease in maximum RAM usage by a factor of 6.6x compared to the original models. These results highlight the efficiency and feasibility of our approach for processing scene text on integer-only embedded hardware.

REFERENCES

[1] D. Cao, Y. Zhong, L. Wang, Y. He, and J. Dang, "Scene text detection in natural images: A review," *Symmetry*, vol. 12, no. 12, p. 1956, Nov. 2020, doi: 10.3390/sym12121956.

[2] X. Li, J. Liu, and S. Zhang, "Text recognition in natural scenes: A review," in *Proc. Int. Conf. Culture-Oriented Sci. Technol. (ICCST)*, Oct. 2020, pp. 154–159, doi: 10.1109/ICCST50977.2020.00036.

[3] B. Zhi-Cheng, L. Qing, C. Peng, and G. Li-Qing, "Text detection in natural scenes: A literature review," *Chin. J. Eng.*, vol. 42, no. 11, pp. 1433–1448, 2020, doi: 10.13374/j.issn2095-9389.2020.03.24.002.

[4] C. Gopalan and D. Manjula, "Sliding window approach based text binarisation from complex textual images," 2010, *arXiv:1003.3654*.

[5] K. Wang and S. J. Belongie, "Word spotting in the wild," in *Proc. Eur. Conf. Comput. Vis.* Glasgow, U.K.: Springer, Sep. 2010, pp. 591–604.

[6] J. Fabrizio, B. Marcotegui, and M. Cord, "Text detection in street level images," *Pattern Anal. Appl.*, vol. 16, pp. 519–533, Nov. 2013.

[7] 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.

[8] Y. C. Wei and C. H. Lin, "A robust video text detection approach using SVM," *Exp. Syst. Appl.*, vol. 39, no. 12, pp. 10832–10840, Sep. 2012.

[9] Y. Zhang, C. Wang, B. Xiao, and C. Shi, "A new method for text verification based on random forests," in *Proc. Int. Conf. Frontiers Handwriting Recognit.*, Sep. 2012, pp. 109–113.

[10] S. M. Hanif and L. Prevost, "Text detection and localization in complex scene images using constrained AdaBoost algorithm," in *Proc. 10th Int. Conf. Document Anal. Recognit.*, Barcelona, Spain, 2009, pp. 1–5.

[11] X. Zhao, K.-H. Lin, Y. Fu, Y. Hu, Y. Liu, and T. S. Huang, "Text from corners: A novel approach to detect text and caption in videos," *IEEE Trans. Image Process.*, vol. 20, no. 3, pp. 790–799, Mar. 2011.

[12] Q. Ye, Q. Huang, W. Gao, and D. Zhao, "Fast and robust text detection in images and video frames," *Image Vis. Comput.*, vol. 23, no. 6, pp. 565–576, Jun. 2005.

[13] W. Wu, X. Chen, and J. Yang, "Detection of text on road signs from video," *IEEE Trans. Intell. Transp. Syst.*, vol. 6, no. 4, pp. 378–390, Dec. 2005.

[14] Y. Zhu, C. Yao, and X. Bai, "Scene text detection and recognition: Recent advances and future trends," *Frontiers Comput. Sci.*, vol. 10, no. 1, pp. 19–36, Feb. 2016.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. NIPS*, vol. 1. Red Hook, NY, USA: Curran Associates, 2012, pp. 1097–1105.

[16] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Reading text in the wild with convolutional neural networks," *Int. J. Comput. Vis.*, vol. 116, pp. 1–20, Jan. 2015.

[17] Z. Zhong, L. Jin, S. Zhang, and Z. Feng, "DeepText: A unified framework for text proposal generation and text detection in natural images," 2016, *arXiv:1605.07314*.

[18] Y. Jiang, X. Zhu, X. Wang, S. Yang, W. Li, H. Wang, P. Fu, and Z. Luo, "R2CNN: Rotational region CNN for orientation robust scene text detection," 2017, *arXiv:1706.09579*.

[19] Z. Tian, W. Huang, T. He, P. He, and Y. Qiao, "Detecting text in natural image with connectionist text proposal network," in *Proc. Eur. Conf. Comput. Vis.* Amsterdam, The Netherlands: Springer, Sep. 2016, pp. 56–72.

[20] D. Xiang, Q. Guo, and Y. Xia, "Robust text detection with vertically-regressed proposal network," in *Proc. Eur. Conf. Comput. Vis.* Amsterdam, The Netherlands: Springer, 2016, pp. 351–363.

[21] Z. Zhang, C. Zhang, W. Shen, C. Yao, W. Liu, and X. Bai, "Multi-oriented text detection with fully convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 4159–4167.

[22] C. Yao, X. Bai, N. Sang, X. Zhou, S. Zhou, and Z. Cao, "Scene text detection via holistic, multi-channel prediction," 2016, *arXiv:1606.09002*.

[23] X. Li, W. Wang, W. Hou, R.-Z. Liu, T. Lu, and J. Yang, "Shape robust text detection with progressive scale expansion network," 2018, *arXiv:1806.02559*.

[24] C. Du, C. Wang, Y. Wang, Z. Feng, and J. Zhang, "TextEdge: Multi-oriented scene text detection via region segmentation and edge classification," in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sep. 2019, pp. 375–380.

[25] X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He, and J. Liang, "EAST: An efficient and accurate scene text detector," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2642–2651, doi: 10.1109/CVPR.2017.283.

[26] K. Wang, B. Babenko, and S. Belongie, "End-to-end scene text recognition," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 1457–1464, doi: 10.1109/ICCV.2011.6126402.

[27] G. Li, "CSNet-PGNet: Algorithm for scene text detection and recognition," in *Proc. 3rd Int. Conf. Comput. Vis., Image Deep Learn. Int. Conf. Comput. Eng. Appl. (CVIDL ICCEA)*, May 2022, pp. 1217–1224, doi: 10.1109/CVIDLICCEA56201.2022.9824815.

[28] A. Khalil, M. Jarrah, M. Al-Ayyoub, and Y. Jararweh, "Text detection and script identification in natural scene images using deep learning," *Comput. Electr. Eng.*, vol. 91, May 2021, Art. no. 107043.

[29] X. Rong, C. Yi, and Y. Tian, "Unambiguous text localization, retrieval, and recognition for cluttered scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 3, pp. 1638–1652, Mar. 2022, doi: 10.1109/TPAMI.2020.3018491.

[30] R. Bagi, T. Dutta, and H. P. Gupta, "Cluttered TextSpotter: An end-to-end trainable light-weight scene text spotter for cluttered environment," *IEEE Access*, vol. 8, pp. 111433–111447, 2020, doi: 10.1109/ACCESS.2020.3002808.

[31] Y. Cao, S. Ma, and H. Pan, "FDTA: Fully convolutional scene text detection with text attention," *IEEE Access*, vol. 8, pp. 155441–155449, 2020, doi: 10.1109/ACCESS.2020.3018784.

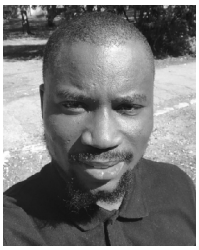
[32] Y. Liu, C. Shen, L. Jin, T. He, P. Chen, C. Liu, and H. Chen, "ABCNet v2: Adaptive Bezier-curve network for real-time end-to-end text spotting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 11, pp. 8048–8064, Nov. 2022, doi: 10.1109/TPAMI.2021.3107437.

[33] J. Guo, R. You, and L. Huang, "Mixed vertical-and-horizontal-text traffic sign detection and recognition for street-level scene," *IEEE Access*, vol. 8, pp. 69413–69425, 2020, doi: 10.1109/ACCESS.2020.2986500.

[34] C. Zhang, Y. Tao, K. Du, W. Ding, B. Wang, J. Liu, and W. Wang, "Character-level street view text spotting based on deep multisegmentation network for smarter autonomous driving," *IEEE Trans. Artif. Intell.*, vol. 3, no. 2, pp. 297–308, Apr. 2022, doi: 10.1109/TAI.2021.3116216.

[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, doi: 10.1109/TIP.2019.2954218.

[36] 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.
[37] C. Zaharia, D. Dinu, and A. Caliman, "PVANet optimization for person detection," in *Proc. Int. Conf. Optim. Electr. Electron. Equip. (OPTIM), Int. Aegean Conf. Electr. Mach. Power Electron. (ACEMP)*, May 2017, pp. 959–964, doi: 10.1109/OPTIM.2017.7975094.
[38] *Google Coral Products Page*. Accessed: Jan. 5, 2023. [Online]. Available: https://coral.ai/docs/dev-board/datasheet/



OLUTOSIN AJIBOLA ADEMOLA received the M.Sc. (Eng.) degree from the School of Information Technology, Tallinn University of Technology, Tallinn, Estonia, in 2020. His research interests include deep learning, machine learning, edge AI, intelligent systems, and computational intelligence.

EDUARD PETLENKOV (Member, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees in computer and systems engineering from the Tallinn University of Technology, in 2001, 2003, and 2007, respectively. He is currently a tenured Full Professor with the Department of Computer Systems, Tallinn University of Technology, and the Head of the Centre for Intelligent Systems. His main research interests include the domain of intelligent control, system analysis, and computational intelligence.

MAIRO LEIER received the Ph.D. degree in computer systems from the Tallinn University of Technology, in 2016. He was a Research Scientist with the Department of Computer Systems, Tallinn University of Technology, where he leads the Embedded AI Research Laboratory. His current research interests include machine learning on embedded systems, optimization techniques, and edge computing.