Received: August 8, 2022. Revised: August 30, 2022. 

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

# Arbitrary Oriented Scene Text Recognizer (AOTR)

**Roopa Mirle Jayanth[1]*** **Mahantesh Kapanaiah[2]**

[1]*Department of Computer Science and Engineering, SJB Institute of Technology, Affiliated to Visveswaraiah Technological University, Karnataka, India*
[2]*Department of Electronic Communication and Engineering, SJB Institute of Technology, Affiliated to Visveswaraiah Technological University, Karnataka, India*
* Corresponding author's Email: mjroopashobith@gmail.com

**Abstract:** Recognizing arbitrary oriented text has grabbed researchers' attention to develop several algorithms due to its high complexity in scene images and real-time applications like language translators, reading text for blind people, navigation systems, and smart parking. Converting text in natural scene images into strings has extended its application to Natural Language Processing (NLP)-based applications like named entity recognition. There are only three methods for text recognition: Optical Character Recognition (OCR), conventional methods, and Neural Network (NN) models. OCR is only known to be a successful text recognizer for scanned images, and in recent years, NN models have outperformed traditional methods for scene text recognition. It is necessary to create an optimal model to address the issue of scene text irregularity. We present the NN and customized OCR models, which we tailor for arbitrarily oriented text recognition, thereby avoiding scene text irregularity. An Orientation Correction Model (OCM) was introduced to improve the recognition model. In place of the recognition model, we used OCR. Alternatively, we created another model that reads corrected text images and extracts low-level features using the convolutional neural network layer. A recurrent neural network then uses these features to recognize text. Experiments were conducted on ICDAR2015, Total Text, Art19, and Cute80 benchmark datasets. It is observed that the proposed model obtains 79.5 % accuracy and hence increases the result by 1.9 % compared to the existing method after adding the orientation correction model. Similarly, results are promising on other datasets compared to existing algorithms.

**Keywords:** Convolution neural network, Recurrent neural network, Arbitrary oriented text, Optical character Recognition.

## 1. Introduction

In recent days, several improved algorithms have been introduced for recognizing different oriented text from natural scene images. Many researchers are still working to find an optimal solution for this problem, which has challenges such as a non-uniform background, non-uniform lighting, blurriness, and different text orientations in the images. As an extension to these irregular text challenges, the comic onomatopoeia dataset [1] is created. Onomatopoeias are the irregular text randomly placed on the images of comic books when exaggerated sound or some different state of the object is to be represented. Recognition of this kind of text irregularity and complex background is impossible by any OCR. OCR is known to be a successful text recognizer only for scanned images. These OCR models have been intensely trained and used in scene text recognition. Unfortunately, commercial OCRs are only apt for horizontal orientated text and uniform background images than un-uniformed background [2]. Resolving issues of the multi-oriented problem of scene text recognition [3-6] can be carried out in three ways. It can be done by introducing traditional algorithms, creating well-trained neural network models, or customizing ready OCR for correcting text orientation [7, 8] to be legible for human and computer systems. In this paper, we proposed two solutions for the above-said problem. We also compared both methods on benchmark datasets and existing methods.


Figure. 1 Example of arbitrary oriented text

In this paper, we attempted to correct different orientated texts and feed them to the recognition model. The simplicity and efficiency of this model greatly impact text recognition improvement. We proposed two models for text recognition. The first model invokes the following function: In model1, text region orientations are corrected and passed as input to the convolutional neural network (CNN) layer to extract low-level features. The next step uses these features and processes them by a recurrent neural network (RNN) to recognize individual characters and words. In model 2, the first function is to find out the orientation of each character in the detected text region and then bring them to the horizontal orientation prior to recognition. This orientation correction makes the recognition process more successful. These corrected images are then given as input to traditional OCR. We used easy OCR for this case.

These functions experimented on ICDAR2015 [9], Total Text [10], Cute80 [11], and Art19 [12] datasets and results are recorded. Sample arbitrary-oriented text images are shown in Fig. 1.

The primary reason for writing this paper is to determine the cause of unsuccessful recognition of arbitrarily placed text on scene images and to present a simple and more efficient strategy for correcting the various orientations of image text. This paper makes two contributions:

The impact of irregularity removal (text orientation correction), along with

i) OCR and

ii)CNN+RNN combination for arbitrarily oriented text recognition.

The rest of the paper includes a Literature survey in the 2nd section, Methodology in the 3rd section, Experimentation in the 4th section, Result discussion in the 5th section, and finally end with the conclusion in the 6th section.

## 2. Literature survey

Over the last few decades, it has been observed that a solution for the recognition of irregular scene text problems would be possible using either a conventional approach, OCR, or neural network approaches. This section presents a few approaches that use the orientation correction model with recognition and a few on direct recognition. All these approaches can be classified as orientation correction with recognition and recognition without text orientation correction model. The pros and cons of all these approaches are also discussed.

### 2.1 Direct recognition methods

The technique used in [13-16] is OCR for scene text recognition but works with only regular and fixed-length text recognition, and it is not suitable for varied text lengths. In [17], instead of using ready OCR for text recognition, a novel approach is introduced to address multi-oriented and bilingual text issues. It introduces graph representation of characters and dynamic programming to reduce large classification problems into a small classification problem. The loss function was also used to increase recognition. However, this worked well with the regional language Kannada as these characters are more curved by nature, but it was not successful in the case of English text as these characters have straight lines than curve shapes. This method failed to give graph representation for the letter 'o' and digit '0'. Another failure case is that it cannot distinguish between the letter 'W' and 'M' and hence could not increase English text recognition.

An approach was framed in [18] that uses a dynamic log-polar transformer to correct arbitrary oriented text. This technique is well suited for multi-scale and multi-oriented text. The drawback of this approach is that it cannot handle highly long or blurry text. CNN+RNN combination network was developed to address the issue of the distorted image [19]. They used an improved residual block to recognize text in any orientation. This performs high-level feature extraction of the text of the input image. When it comes to the perspective text, this performs worse than [20]. Two models- Character Anchoring and Anchor Pooling- are combined to create feature sequences and extract high-level semantics from two-dimensional space [21] to address the bounding box techniques [6] issues for arbitrary oriented text. The creator of mask text spotter [22] used the semantic segmentation technique to recognize oriented text successfully. They also used a fully convolutional network for recognition. However, it is a bit slower than others' work.

### 2.2 Orientation correction with recognition methods

In [23], they introduced a method for handling all categories of vertical texts and a new dataset that consists of all variants of vertical texts. They began by extracting features, which they then fed into an RNN and a connectionist temporal classification (CTC) to predict text. They addressed only vertical text, neither curved nor any other irregularity in the text. An end-to-end text recognizer [24] handles text irregularities using a thin-plate transformation mechanism and using control point prediction; they present a rectification model. They use the spatial transformer network (STN) framework. In [25], they adopted the Segmentation Proposal Network (SPN) for anchor-free representation for arbitrary oriented text instead of region proposal networks (RPN). Further masked Region of Interest (RoI) features are used to detach every text instance from its neighboring text to increase the recognition rate. In [26], initially, they introduced a rectification model to correct arbitrary oriented text. For recognition, they used a pre-trained standard ResNet model with FPN to get text features and RNN for sequence learning. The thin plate spline transformation method is used for automatic rectification [20] that handles text irregularities. Curved text irregularity and perspective text recognition were accomplished using a spatial transformer and a sequence recognition network. However, it fails in the case of the largely curved text. This literature review confirms that oriented correction models perform better than direct recognition models.

## 3. Methodology

This section elaborates on the detailed functioning of proposed recognition models. The text recognition model assumes that text detected from the detection model is successfully done. Model1 and model2 are the frameworks we proposed for successful recognition. We need to adopt a popular framework to make the recognition model more successful. One of the popular frameworks is CNN+RNN for text recognition. In this paper, we used the same framework as model1. However, the layer numbers are defined as per our requirements. A combination of CNN and RNN can handle sequences in arbitrary lengths without an additional lexicon. Here CNN is used to extract high-level features of the text. Furthermore, RNN is used for sequence learning. This combination of models is best suited for unknown words, arbitrary strings, and multiple words without any lexicon. Model1 and model2 are the frameworks we proposed for successful recognition.

**Model 1:** In the following figure, the sample dataset image used in our work is presented as an input image and annotations of each character and word for the input image and its equivalent josn file. Refer Fig. 2.


(a)

"name":"polygon","all_points_x":[141,79,110,167],"all_points_y":[289,312,401,374]},"region_attributes":{"Name2":"S"}},
"shape_attributes":{"name":"polygon","all_points_x":[163,217,232,179],"all_points_y":
277,255,340,361]},"region_attributes":{"Name2":"E"}},{"shape_attributes":{"name":"polygon","all_points_x":
232,305,319,236],"all_points_y":[252,233,318,340]},"region_attributes":{"Name2":"A"}},{"shape_attributes":
"name":"polygon","all_points_x":[318,405,405,330],"all_points_y":[224,212,305,318]},"region_attributes":{"Name2":"C"}},
"shape_attributes":{"name":"polygon","all_points_x":[421,500,495,416],"all_points_y":
205,199,302,305]},"region_attributes":{"Name2":"R"}},{"shape_attributes":{"name":"polygon","all_points_x":
518,586,568,506],"all_points_y":[200,204,305,301]},"region_attributes":{"Name2":"E"}},{"shape_attributes":
"name":"polygon","all_points_x":[617,680,647,588],"all_points_y":[206,219,317,306]},"region_attributes":{"Name2":"S"}},
"shape_attributes":{"name":"polygon","all_points_x":[692,781,738,657],"all_points_y":
216,243,345,317]},"region_attributes":{"Name2":"T"}},{"shape_attributes":{"name":"polygon","all_points_x":
177,246,218,151],"all_points_y":[501,519,620,600]},"region_attributes":{"Name2":"B"}},{"shape_attributes":
"name":"polygon","all_points_x":[266,322,308,250],"all_points_y":[521,525,634,628]},"region_attributes":{"Name2":"E"}},
"shape_attributes":{"name":"polygon","all_points_x":[339,338,411,421,329],"all_points_y":
528,529,530,633,633]},"region_attributes":{"Name2":"A"}},{"shape_attributes":{"name":"polygon","all_points_x":
430,519,536,451],"all_points_y":[530,511,619,638]},"region_attributes":{"Name2":"C"}},{"shape_attributes":
"name":"polygon","all_points_x":[545,628,655,567],"all_points_y":[503,477,591,618]},"region_attributes":{"Name2":"H"}},
"shape_attributes":{"name":"polygon","all_points_x":
78,154,223,308,410,500,599,680,779,736,651,578,496,407,322,236,171,107],"all_points_y":
316,282,257,230,208,199,202,218,243,343,316,304,301,304,317,336,364,401]},"region_attributes":{"Name2":"SEACREST"}},
"shape_attributes":{"name":"polygon","all_points_x":[172,173,254,330,424,534,627,655,546,433,317,230,151],"all_points_y"
494,492,521,528,532,508,478,589,623,633,633,620,600]},"region attributes":{"Name2":"BEACH"}}],"file attributes":{}}}

(b)

Figure. 2: (a) Equivalent .json file for annotated image (right image) and (b) Annotated image and its .json file used for scene text recognition

(a) (b) (c)

Figure. 3 Sample training and testing data: (a) Individual character for the training phase, (b) Word samples for the training phase and (c) Input image for proposed model – Testing data

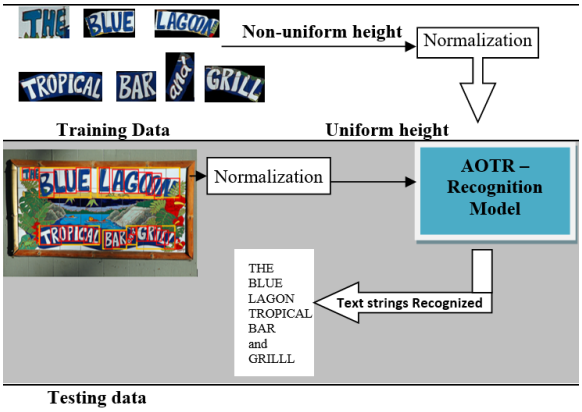



Figure. 4 Workflow of scene text recognition-model 1

Fig. 3 shows a sample of train and test data used in this model. Fig.4 represents the workflow of scene text recognition. This takes the input for training and testing which are of the form shown in Fig. 3. All Sample inputs are normalized to get a uniform height. Normalized inputs are given to the AOTR model for further processing. AOTR recognition model converts the input image into text strings.

After successful detection from any text detection model, the cropped text regions are sent through Convolutional layers to, along with pooling layers, obtain the features of the image. And then, these features are given to Long Short-Term Memory (LSTM) architecture, which is many-to-many, and it outputs SoftMax probabilities over the vocabulary. These outputs are given to a dense layer to obtain raw text strings from the image.

### 3.1 Model functioning

- Input:

  Image of size (256, 64) is passed as input to the model.
- Convolutional Layer:
  - A Convolutional layer with 32 nodes is used to extract features from the input layer. Batch normalization is added after the Convolutional layer to normalize the input data while training. Relu activation is used here. Max pooling is done after normalization.
  - 3 layers of Convolutional layers are used in the model. After the input is passed through 3 layers of CNN, and it is then passed to 2 layers of RNN (LSTM layers).
- Recurrent Network Layer:
  - The image is reshaped to size (64, 1024) and passed to a dense layer to change the shape of the input vector.
- Bidirectional LSTM Layers:
  - LSTM layers are used to improve the model training and performance for classification problems. 2 layers of LSTM with 256 nodes are used in the model.
- Dense layer:
  - Finally, a dense layer of 64 nodes for each character to be recognized is added to give the final output.

### 3.2 Model 1 structure

Convolution and Bi-LSTM layers are combined in the design of model1 for text recognition in any orientation. The architecture of CNN and RNN for arbitrarily oriented text recognition is shown in Fig. 5. The layers in this framework are as follows: an input layer, three times convolution layers, batch normalization, and max-pooling, dropout layer after 2nd and third pooling, dense layers with two times LSTM, and a Softmax layer.

Input layer

### 3.3 Convolution layer

The input data is represented as low-level features by the convolution layer. The patterns of an image, including edge and texture details of different regions of the objects, are detected, and interpreted in this layer using static size filters. The process produces the features needed to identify and categorize the entire object in an image. In this model, all three level convolutions use 3x3 size filters. But level 1 convolution uses 32 filters, in level 2,64 filters, and in level 3,128 filters are used.

The input layer reads an input image in terms of a two-dimensional array of numbers corresponding to its pixels. Each input image is of size 256 x 64. Fig. 3 depicts input character samples and word examples.

Between the convolutional and ReLu layers, batch normalization layers are added to speed up training, reduce network initialization sensitivity, improve the activation function's viability, and achieve deeper network simplification. Mathematically convolution is represented with Eq. (1).

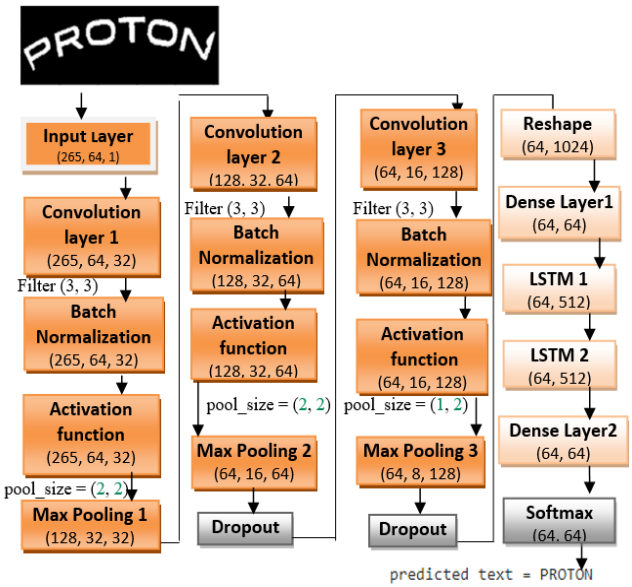



Figure. 5 Network architecture of the proposed model 1 for arbitrary oriented text

$$z = x * y \tag{1}$$

Where x represents the input, y represents the filter, and z is the convolution.

#### 3.3.1. ReLU

Rectified Linear Unit (ReLU) is a commonly used non-linear activation function in many network architectures. Because it allows the model to learn fast and perform better than other activation functions. It applies the function 'f' shown in Eq. (2) for each input pixel and hence converts negative value pixels to zero.

$$f(i) = max(0, 0, i) \tag{2}$$

Where '$i$' is input, f is the function.

### 3.4 Pooling layer

These layers are mainly used to reduce feature map dimensions. It controls the overfitting of the data, thereby reducing the recognition error. In CNN architectures, pooling layers are intermediate layers inserted between the subsequent convolutional layers. Max-pooling is the most commonly used pooling function which returns the maximum value from the feature map. The pooling layer provides another form of rotation invariance by taking the most salient information from the local region. This process outputs prominent feature values from the given image. For the given feature map ($f_{fm}$) of the size shown in Eq. (3), the pooling layer produces output ($f_{pl}$) of the size shown in Eq. (4).

$$f_{fm} = f_h * f_w * f_c \tag{3}$$

$$f_{pl} = (f_h - f_s + 1)/s * (f_w - f_s + 1)/s * f_c \tag{4}$$

Where $f_h$ is the height, $f_w$ is the width, $f_c$ is the channels number. $f_s$ filter size, and 's' is the stride length. After pooling layers 2 and 3, a dropout layer is used to optimize the model. To prevent overfitting of training data, the dropout layer ignores some of the features.

### 3.5 Bidirectional long short-term memory (Bi-LSTM)

Long short-term memory networks, abbreviated as LSTM, are a type of RNN. They were introduced to avoid the problem of long-term dependency, and LSTM networks have a similar structure to the RNN. Bi-LSTM is the process of making any neural network that can store sequence information in both directions, forward and backward. Our input flows in bidirectional mode, distinguishing Bi-LSTM from a regular LSTM.

Because the BiLSTM model is a hybrid of forward and backward LSTM networks, it can read input reviews from both directions. The forward LSTM and backward LSTM process information from left to right and right to left, respectively. Eqs. (5) and (6) represent the hidden state (hs) of forward LSTM and forward LSTM, respectively.

$$\overrightarrow{hs_i} = LSTM\left(x_i, \overrightarrow{hs_{i-1}}\right) \tag{5}$$

$$\overleftarrow{hs_i} = LSTM\left(x_i, \overleftarrow{hs_{i+1}}\right) \tag{6}$$

$$hs = \left[\overrightarrow{hs_i}, \overleftarrow{hs_i}\right] \tag{7}$$

### 3.6 Dense layer

The dense layer is the last in CNN and is often a fully connected layer, which connects every neuron with every element of the previous layer. These layers are similar to those in regular Multi-Layer Perceptron networks and output the final predictions of various visual object recognition problems. The inputs to the dense layer are considered feature value outcomes obtained after applying the three levels of convolution and two levels of Bi-LSTM. It is a learning layer, as in a conventional neural network. In error propagation, the output layer of a convolution neural network back propagates to initiate re-learning from the fully connected layer. General function used in the dense layer is represented as shown in Eq. (7)

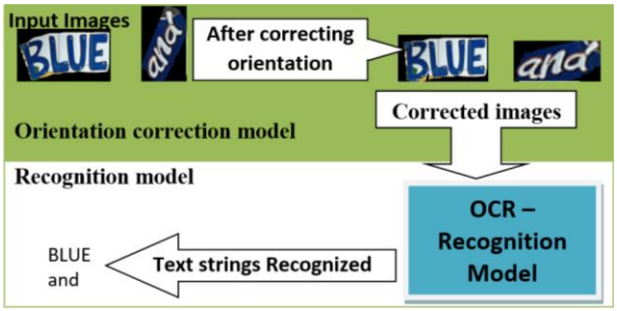



Figure. 6 Workflow of scene text recognition-model2

$$o = activatioin\left((i \,.\, k) + b\right) \quad (7)$$

Where 'o' is the output, 'i' is the input, 'k' is the kernel, and 'b' is the bias

### 3.7 Model 2:

In this model, we introduce the orientation correction model as pre-processing step. Fig. 6 shows how recognition and text orientation correction models were connected for successful recognition tasks.

### 3.8 Orientation correction model (OCM)

Arbitrarily oriented text may not be perpendicular to the camera, i.e., perspective text or non-perspective means; texts from different angles oriented or curved text. Orientation corrections must be made to segment these texts from the image's background. Precise boxes taken from detection mode around each character and words are considered as input to the orientation correction model. The minimum area rectangle function (refer to Fig. 7) is used to find the angle of the oriented box with the positive or negative x-axis, respectively. The values of each 2d object (oriented boxes in this paper) are as follows: the top-left corner of the oriented box is denoted as (x, y), the width of the oriented box, the height of the oriented box, and its angle θ of rotation to the x-axis (refer Eq. (9)).

$$object = ((x, y), w, h, \theta) \quad (9)$$

Where $'w'$ is the with, $'h'$ is the height, and 'θ' is the angle of the bounding box.

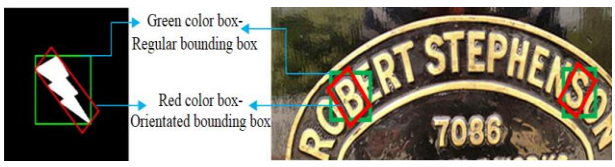



Figure. 7 General bounding box obtained (green color) from detection model and oriented bounding box (red color)

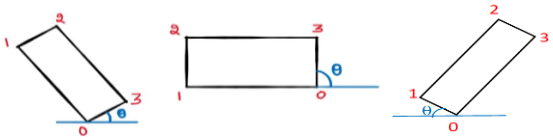


Figure. 8 Angle of rotation

Each bounding box obtained from the minimum area rectangle function is skewed by the angleθ. Bounding box corners are labeled from highest y as initial point 0 and continue in clockwise direction finally end point labeled as 3 (refer to Fig. 8). If the orientation is in the first quadrant, an angle θ is computed between corners 0 and 1, and correction is done in the left direction. If the orientation is in the second quadrant, an angle θ is computed between corners 0 and 3, and the correction box is done in the right direction.

### 3.9 Recognition model

EasyOCR, unlike Tesseract, can use OpenCV's default BGR color channel ordering. As a result, after loading the image, we do not need to swap color channels. In this, EasyOCR is used for the recognition task. It has been proved that this OCR is more genuine when the input is in regular text. Hence this section connects the orientation model to this model. EasyOCR employs ResNet as a foundation framework with long short-term memory (LSTM). It employs the CTC method to handle sequence problems in recognition. Results from various datasets are obtained by combining the OCM with the recognition model. These results are presented in the IV section.

## 4. Experiments

The datasets and results from the proposed model's comparison with current best practices both with and without its OCM are discussed in the following subsections.

### 4.1 Datasets

We used four Benchmark datasets to assess the proposed model: ICDAR15, CUTE80, Art19, and TotalText.

### 4.2 ICDAR2015 [9]

The ICDAR2015 dataset is includes for training 1,000 images and for testing 500 images generated by Google glasses. It has horizontal English text, varied scales of text, blurred text, and orientation text.

### 4.3 Total text [10]

Total-Text focuses more on multioriented and curve text. Out of total text, 1255 images are used for training and 300 images used for testing, and it provides word-level polygon annotations along with transcription.

### 4.4 CUTE80 [11]

This dataset has curved text. It includes 80 images which are taken from high-resolution from natural scenes. CUTE80 is proposed for detection tasks and does not contain lexicon. But we annotated all the words and set of 300 images are used for testing in recognition task.

### 4.5 Art19 [12]

With diversity of text in mind this dataset is created by choosing only images focuses on arbitrary oriented text from TotalText, SCUT-GW1500 and Baidu curved scene text benchmark datasets. It contains 10166 total images, 5603 for training and 4563 for testing respectively.

## 5. Results

We evaluate our proposed models on various standard datasets for their performance. Two challenging tasks, such as regular and irregular text recognition, are involved in datasets. Some of the datasets we used include CUTE80, Art19, ICDAR15, and TotalText.

Table 1 and 2 show that when OCMs are considered, model2 produces better results than model1. This is because model2 employs a pretrained model, which yields superior results. When the OCM is not used, model1 outperforms model2. On the CUTE80 dataset, [27] depends on a dictionary of words and cannot perform well when a new word appears on the image. [28] cannot successfully recognize curved text. These two methods try to recognize detected text directly; thus, the recognition rate is low compared to others. The model used in [20] rectifies curved text before recognition but still, it is not suitable for largely cured text as it the uses STN method which doesn't rectify text perfectly. But model is more reliable because we corrected all orientation of the text and made them horizontal using OCM and then recognition task is applied. Hence able to achieve better accuracy. The effectiveness of the largely oriented text recognition algorithms is assessed using benchmark datasets like ICDAR2015, CUTE80, Art19, and Total text. Curve text is a frequently encountered artistic-style text in natural scenes, and because of its irregular character placement, it is very difficult to identify. Using the unconstrained-lexicon method, we compare models1 and 2 for perspective and curved text recognition in order to validate the OCM. Largely oriented text samples cause less recognition and reduce its rate in

Table 1. Recognition accuracies on benchmark datasets CUTE80 and ICDAR15. Result comparison with state-of-the-art technique

| Model | CUTE80 | | Model | ICDAR15 |
|---|---|---|---|---|
| Jaderberg et al. [27] | 42.7 | | TextNet [6] | 60.5 |
| Shi et al. [28] | 54.9 | | Mask TextSpotter [22] | 62.4 |
| RARE [20] | 59.2 | | Wei [3] | 63.3 |
| RNTR-Net [19] | 62.6 | | ASTER [24] | 67.6 |
| Wei [3] | 67.4 | | ASTER [24]+Rectification | 68.9 |
| ASTER [24] | 73.26 | | Li et al. [29] | 69.2 |
| ASTER [24]+Rectification | 76.39 | | Text Perceptron [30] | 67.9 |
| Model 1 | 72.01 | | Model 1 | 73.2 |
| Model 1 + OCM | 76.62 | | Model 1 + OCM | 77.1 |
| Model 2 | 71.05 | | Model 2 | 75.8 |
| Model 2+ OCM | 78.20 | | Model 2+ OCM | **78.5** |

Table 2. Recognition accuracies on benchmark datasets TotalText and Art19. Result comparison with state-of-the-art technique

| Model | Total Text | | Model | Art19 |
|---|---|---|---|---|
| Long [26] | 76.3 | | Long [26] | 72.1 |
| Long [26]+Rectify | 77.6 | | Long [26]+Rectify | 72.3 |
| Model 1 | 71.4 | | Model 1 | 69.8 |
| Model 1+ OCM | 75.2 | | Model 1+ OCM | 72.6 |
| Model 2 | 70.4 | | Model 2 | 67.6 |
| Model 2+ OCM | **79.5** | | Model 2+ OCM | **73.8** |

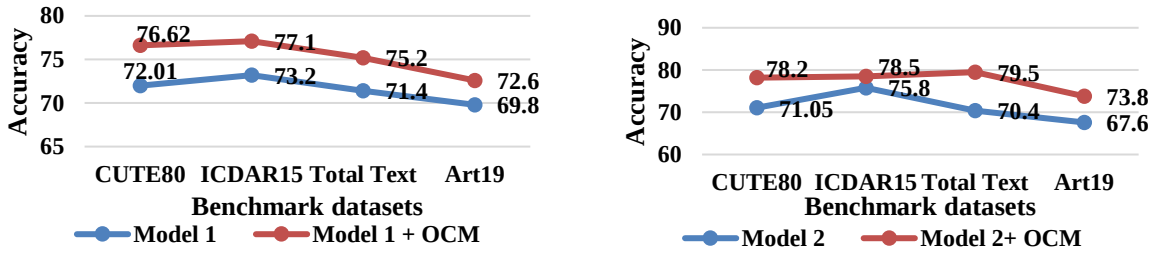
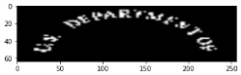
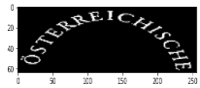



Figure. 9 Results comparison: (a) Model 1 and (b) Model 2 results comparison with its improved version by adding OCM respectively

Table 3. Sample output of the proposed model with successful cases and failure cases

| | True Positive samples | | True Negative samples | |
|---|---|---|---|---|
| Text Orientation type | Sample input | Predicted text from the Model1 &2 +OCM | Sample input | Predicted text from the Model1&2+OCM |
| Perspective |  | WALK |  | FHUNTER |
| curved |  | SINGAPORE |  | fractured |
| Largely curved |  | USDEPARTMENTOF |  | KADVENTURE |
| Horizontal |  | adult | | |
| Perspective |  | LAND | | |
| Largely curved |  | OSTEREICHISCHE | | |
| curved |  | Ambrosi | | |
| Perspective |  | CLUB | | |
| Largely curved |  | BIENVENUE | | |

several methods of Literature. The results of the proposed models are compared with the performance of other approaches on four benchmark datasets. All the recognition accuracies on the four datasets obtained by our models include two parts: with OCM and without OCM.

Fig. 9 shows that after adding the orientation correction model to both models1 and 2, the accuracy of both models increases by about 2.7% to 9.1% on different datasets. Table 3 displays the output of the proposed model for various text orientation types such as perspective, curved, and largely curved, which are the more difficult styles of text. Our modified models are efficient enough to produce correct text strings as output.

## 6. Conclusion

In this paper, we determined the cause of unsuccessful recognition of arbitrarily placed text on scene images. Correcting the angle of any orientated text is obtained by OCM and made horizontal text, then the recognition task is performed. The impact of the irregularity removal (text orientation correction model) on the OCR and Neural network model was explored for arbitrarily oriented text recognition.

In this paper, combined neural networks and customized OCR models are suggested for recognizing text from natural scene images. The model1 showcases CNN with RNN integration. CNN can read images with non-uniform input, extract features, and make predictions. A dense layer is used for predicted text strings, and bidirectional LSTM layers are used for sequence recognition labeling. The performance of the OCM with OCR on benchmark datasets is compared to that of CNN+RNN-based algorithms and other cutting-edge techniques. The irregularity of text is a frequent and prominent issue in natural scene image text recognition that is addressed. Any oriented text can be handled by the

method and converted into horizontally oriented text. By doing this, we can raise the rate of recognition. Furthermore, the model1 can be trained for many samples with a proper supporting environment. This work only contributes toward the multi-oriented text and can be extended to multilingual text recognition by incorporating a script identification model.

## Conflicts of Interest

The authors declare no conflict of interest.

## Author Contributions

The paper background work, conceptualization, methodology, dataset collection, implementation, results analysis, and comparison, preparing and editing of the draft, and visualization has been done by the first author. The supervision, review of work, and project administration have been done by the second author.

## References

[1] J. Baek, Y. Matsui, and K. Aizawa, "COO: Comic Onomatopoeia Dataset for Recognizing Arbitrary or Truncated Texts", In: *Proc. of European Conference on Computer Vision, Tel Aviv*, 2022.

[2] H. Li, P. Wang, and C. Shen, "Towards End-to-End Text Spotting with Convolutional Recurrent Neural Networks", In: *Proc. of IEEE International Conference on Computer Vision (ICCV-2017)*, pp. 5248-5256, 2017.

[3] G. Wei, W. Rong, Y. Liang, X. Xiao, and X. Liu, "Toward Arbitrary-Shaped Text Spotting Based On End-To-End", *IEEE Access*, Vol. 8, pp. 159906-159914, 2020.

[4] T. He, Z. Tian, W. Huang, C. Shen, Y. Qiao, and C. Sun, "An End-to-End TextSpotter with Explicit Alignment and Attention", In: *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR-2018)*, pp. 5020-5029, 2018.

[5] X. Liu, D. Liang, S. Yan, D. Chen, Y. Qiao, and J. Yan, "FOTS: Fast Oriented Text Spotting with a Unified Network", In: *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR-2018)*, pp. 5676-5685, 2018.

[6] Y. Sun, C. Zhang, Z. Huang, J. Liu, J. Han, E. Ding, "TextNet: Irregular Text Reading from Images with an End-to-End Trainable Network", In: *Proc. of Computer Vision ACCV-2018, Proc. of 14th Asian Conference on Computer Vision*, pp. 83-99, 2019.

[7] J. Ma, W. Shao, H. Ye, L. Wang, H. Wang, Y. Zheng, and X. Xue, "Arbitrary-oriented scene text detection via rotation proposals", *IEEE Transactions on Multimedia*, Vol. 20, No. 11, pp. 3111-3122, 2018.

[8] N. Reddy and P. S. Deshpande, "A novel local skew correction and segmentation approach for printed multilingual Indian documents", *Alexandria Engineering Journal*, Vol. 57, No. 3, pp. 1609-1618, 2018.

[9] D. Karatzas, L. G. 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. of 13th International Conference on Document Analysis and Recognition*, pp. 1156-1160, 2015.

[10] C. K. Ch'ng and C. S. Chan, "Total-Text: A Comprehensive Dataset for Scene Text Detection and Recognition", In: *Proc. of 14th International Conference on Document Analysis and Recognition*, Vol. 1, pp. 935-942, 2017.

[11] A. Risnumawan, P. Shivakumara, C. S. Chan, and C. L. Tan, "A robust arbitrary text detection system for natural scene images", *Expert Systems with Applications*, Vol. 41, No. 18, pp. 8027-8048, 2014.

[12] C. K. Ch'ng, Y. Liu, Y. Sun, C. C. Ng, C. Luo, Z. Ni, C. Fang, S. Zhang, J. Han, E. Ding, J. Liu, D. Karatzas, C. S. Chan, and L. Jin, "ICDAR 2019 Robust Reading Challenge on Arbitrary-Shaped Text - RRC-ArT", In: *Proc. of International Conference on Document Analysis and Recognition*, pp. 1571-1576, 2019.

[13] J. Almazán, A. Gordo, A. Fornés, and E. Valveny, "Word Spotting and Recognition with Embedded Attributes", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 36, No. 12, pp. 2552-2566, 2014.

[14] A. Gordo, "Supervised mid-level features for word image representation", In: *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR-2015)*, pp. 2956-2964, 2015.

[15] L. Neumann and J. Matas, "Real-time scene text localization and recognition", In: *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR-2012)*, pp. 3538-3545, 2012.

[16] J. A. R. Serrano, A. Gordo, and F. Perronnin, "Label Embedding; A Frugal Baseline for Text Recognition", *International Journal of Computer Vision*, Vol. 113, No. 3, pp. 193-207, 2015.

[17] M. J. Roopa and K. Mahantesh, "Classification and Recognition of Bilingual Text Using Graph Edit Distance Based Degree of Similarity",

*Indian Journal of Science and Technology*, Vol. 15, No. 27, pp. 1336-1343, 2022.

[18] P. Dai, H. Zhang, and X. Cao, "SLOAN: Scale-Adaptive Orientation Attention Network for Scene Text Recognition", *IEEE Transactions on Image Processing*, Vol. 30, pp. 1687-1701, 2021.

[19] Q. Liang, S. Xiang, Y. Wang, W. Sun, and D. Zhang, "RNTR-Net: A Robust Natural Text Recognition Network", *IEEE Access*, Vol. 8, pp. 7719-7730, 2020.

[20] B. Shi, X. Wang, P. Lyu, C. Yao, and X. Bai, "Robust Scene Text Recognition with Automatic Rectification", In: *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR-2016)*, pp. 4168-4176, 2016.

[21] S. Long, Y. Guan, K. Bian, and C. Yao, "A New Perspective for Flexible Feature Gathering in Scene Text Recognition via Character Anchor Pooling. ICASSP", In: *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing – (ICASSP 2020)*, pp. 2458-2462, 2020.

[22] 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", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 43, No. 2, pp. 532-548, 2021.

[23] O. Y. Ling, L. B. Theng, A. C. Weiyen, and C. Mccarthy, "Development of Vertical Text Interpreter for Natural Scene Images", *IEEE Access*, Vol. 9, pp. 144341-144351, 2021.

[24] B. Shi, M. Yang, X. Wang, P. Lyu, C. Yao, and X. Bai, "ASTER: An Attentional Scene Text Recognizer with Flexible Rectification", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 41, No. 9, pp. 2035-2048, 2019.

[25] M. Liao, G. Pang, J. Huang, T. Hassner, and X. Bai, "Mask TextSpotter v3: Segmentation Proposal Network for Robust Scene Text Spotting", In: *Proc. of Computer Vision ECCV-2020, Proc. of 16th European Conference on Computer Vision*, pp. 706-722, 2020.

[26] S. Long, Y. Guan, B. Wang, K. Bian, and C. Yao, "Rethinking Irregular Scene Text Recognition", In: *Proc. of International Conference on Document Analysis and Recognition, ArT Recognition Track*, 2019.

[27] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Reading text in the wild with convolutional neural networks", *International Journal of Computer Vision*, Vol. 116, No. 1, pp. 1-20, 2016.

[28] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 39, No. 11, pp. 2298-2304, 2015.

[29] H. Li, P. Wang, C. Shen, and G. Zhang, "Show, attend and read: A simple and strong baseline for irregular text recognition", In: *Proc. of 33rd AAAI'19: AAAI Conference on Artificial Intelligence*, pp. 8610-8617, 2019.

[30] L. Qiao, S. Tang, Z. Cheng, Y. Xu, Y. Niu, S. Pu, and F. Wu, "Text Perceptron: Towards End-to-End Arbitrary-Shaped Text Spotting", In: *Proc. of 34th AAAI Conference on Artificial Intelligence (AAAI-20)*, pp. 11899-11907, 2020.