Rotation-Invariant Scene Text Extraction using Transformer Networks

Vanitha Sivagami Sivasankaravel[1*], Sreenivasan S.R[2] and Manoj Rajamohan[3]

[1*]Computer Science and Engineering, Mepco Schlenk Engineering College, Sivakasi, 626005, Tamil Nadu, India.
[2]Computer Science and Engineering, Mepco Schlenk Engineering College, Sivakasi, 626005, Tamil Nadu, India.
[3]Computer Science and Engineering, Mepco Schlenk Engineering College, Sivakasi, 626005, Tamil Nadu, India.

*Corresponding author(s). E-mail(s): svanitha@mepcoeng.ac.in;
Contributing authors: sreenivasan1808@gmail.com;
masterrajamohan@gmail.com;

Abstract

A novel approach to scene text recognition is presented in this work, utilizing a transformer-based framework that enhances accuracy by addressing geometric distortions and background noise. Various forms of text, such as those found on logos, signs, banners, advertisements, and more, are encountered in our daily lives. These texts are not always displayed horizontally, which becomes increasingly important when images of such texts are captured. The overall view can be significantly altered by the perspective from which an image is taken. Furthermore, text may appear against complex backgrounds filled with clutter, irregularities, and bumps, making accurate extraction a considerable challenge for computers. The proposed model operates in two phases: text detection and text recognition. Text detection involves the identification of text regions within the provided image, while text recognition focuses on the actual extraction of words from these detected regions. To address these challenges, deep learning techniques are employed in this project to improve the efficiency and accuracy of scene text recognition. Unlike traditional systems, the proposed approach is tailored for unstructured and real-world scenarios. The transformer-based architecture allows for parallel processing, significantly reducing inference time compared to conventional sequential methods. Additionally, the model is designed to adapt to various text formats, enhancing its robustness across different real-world applications.

Keywords: Scene Text Recognition, Deep Learning, Transformers

1 Introduction

In the real world, we encounter numerous words in various forms, such as on logos, signs, banners, advertisements, and more. These texts are not always presented horizontally, and the distortion becomes even more pronounced when images of such texts are captured. The perspective of the image changes depending on the position from which it is taken. Additionally, texts may appear against complex backgrounds filled with clutter, irregularities, and bumps, making them more challenging for computers to accurately extract. Scene Text Recognition (STR) is a branch of Optical Character Recognition (OCR) focused on identifying and reading text embedded in natural scenes, such as street signs, billboards, license plates, product labels, or any environment outside of printed documents. Unlike traditional OCR, which deals with clean, structured text (e.g., printed pages), STR presents unique challenges due to variations in lighting, angles, distortions, backgrounds, and fonts. Applications like autonomous vehicle navigation and content-based image retrieval need to understand the texts present in these scene images. Advancement in deep neural networks has accelerated the progress in text recognition from scenes in clear images and mildly distorted images. Despite these advancements text recognition from images that are degraded, with complex background or where the text is geometrically distorted still prove a significant challenge.

Existing scene text recognition techniques either use an encoder-decoder architecture or a segmentation approach. In the encoder-decoder approach, the images are encoded into embeddings representing the features with are decoded into characters sequentially using a sequential decoder. The sequential decoder predicts a single character at every time step by giving attention to the features of that character. This methodology fails to accurately predict the characters when the images have complex background clutter and geometric distortions. The segmentation method works by detecting individual character and then form a word from these characters. But such approach requires character-level annotations of the character's position and label during training which is difficult and expensive to collect. Grouping of characters into words is also prone to errors due to incorrect character predictions and possibility of occurrence of redundant characters.

Transformers[1] initially introduced for Natural Language Processing tasks was later modified for Computer Vision tasks[2], [3]. The transformer networks uses self-attention mechanism as the primary learning mechanism. Unlike a Convolutional Neural Network which focus on local patterns using convolution operations, self-attention captures global dependencies between different parts of the image by weighing the relative importance of the parts. Self-attention allows parallel processing of all the tokens making the training efficient. A transformer comprises of an encoder and a decoder. The encoder transforms the input data into a context aware representation. This latent representation captures the important features of the input image

while discarding irrelevant details thereby essentially reducing the dimension of the input. This allows for easier processing. A decoder takes the latent representation produced by the encoder and produces output tokens one by one. In our case it is the text in the image.

We present a two-phase model using transformer networks that doesn't require character level annotations unlike segmentation methods and overcomes the challenges of segmentation methods. The first phase of the model involves detecting the position of the text in the image based on various alignments and visual features using only the ground truth word-level bounding box data as opposed to character level annotations. The second phase of the methodology uses the detected text regions from the first phase to extract characters and form a word. It learns to predict the sequence of characters in the image and arrange them in proper order using only the ground truth text and not character level annotations.

2 Related works

Scene text recognition has been studied widely for years using techniques like sequential decoding and segmentation. Shivakumara et.al proposed a way to detect text candidate points by extracting gradient information from the frequency domain using FFT and used combination of Unet++ and EfficientNet to detect text from images resulted in faster compute times compared to other backbones [4]. However severely degraded image proved challenging for this methodology. Ma et.al [5] upscaled degraded images while simaltaneously improving the text recognition using CRNN and Residual blocks. The performance of the model was lacking for real world datasets which includes distorted images where the texts could be occluded. In order to tackle the limitations of sequential decoding Zhong et.al [6] proposed a nonlinear decoding method to decode characters from a vision transformer encoded latent embeddings. Although it achieved state of the art performance in predicting text from occluded and low quality images, it couldn't predict accurately for images containing invisible and incomplete text. Scenes with complex or decorative background and text that are deformed posed a significant challenge for these models. A GAN using Hilbert transform and Optimum phase congruency (OPC) introduced by Banerjee et.al [7] solved this problem by detecting text from tattoos which are known to have complex and decorative text with deformable skin as background. However the method failed to detect the text when the shape and structure of characters is distorted. The CNN and LSTM based model proposed in [8] suffers from the same problem as other techniques using sequential decoder that it fails for complex background and vertical texts.

Scene images can contain text in various alignments and orientations. Recognizing these texts are a significant challenge. Zhang et.al introduced a novel reading-order estimation module in their Inverse-Like Antagonistic Scene Text Spotting framework that spots inverted texts without sacrificing performance on general ones. Inspite of significant performance increase in inverted texts, the model struggles with special cases like mirrored texts, blurred texts and occluded texts. Wu et.al [9] introduced an attention based model to handle texts with geometric constraints and blurred texts. It consisted of two networks: two-level rectified network which rectified the

geometric distortions and attention-based recognition network, recognized the text. It achieved state-of-the-art performance on slightly distorted texts but struggled with completely curved texts. In [10] a novel approach to fit arbitrarily shaped text was presented by Liu et.al. using adaptive bezier curves in their ABCNet v2. The authors designed a BezierAlign layer to extract features from CNN on arbitrarily shaped text. Cao et.al [11] presented another method to detect arbitrarily shaped texts. It worked based on segmentation and self-attention. The HGR-Net [12] of Bi et.al. combined the top-down approach of contour regression and bottom-up approach of sub-text component extraction for text detection of arbitrary images. HGR-Net used graph based network to learn the global contextual information and position information. In cases where a text had small font and short words, the accuracy was poor. It also suffered from vanishing gradient problem. The methods discussed so far perform poorly for real-time detections. To overcome this challenge Yang et.al [13] introduced CM-Net based on concentric mask that also handles arbitrarily shaped texts. It achieved better performance than [10]. [14] represented the text as a kernel that changed the text line as central area. This allowed them to accurately describe arbitrarily shaped scene texts which was classified using a Convolution Neural Network which helped them achieve state of the art performance in inference speed while maintaining competitive accuracy. Wang et.al. [15] developed a region aware context module to extract region aware text information in local area and a progressive fusion module both of which could be incorporated into any scene text detection models to improve detection performance. [16] focussed mainly on segmentation of handwritten text images for Indian languages. [17] proposed an end-to-end scene text extraction method using CRNN architecture which achieved comptetive performance.

Many methods require a large amount of labeled data, making the data collection time-consuming and expensive. To alleviate this dependency Zhou et.al [18] generated pseudo labels for different scales by dividing the image into multiple regions like background, text and uncertain regions. This pseudo-label generation algorithm along with a scale aware loss help extract different scale texts. The end-to-end model presented in [19] eliminated the intermediary processes like image cropping, feature recalculation, word separation, etc. Although the model was capable of detecting text of arbitrary shapes, the performance was less than desirable in cases where texts are small in size, have uncommon fonts or blurred. [20] used Contrastive Language-Image Pretraining (CLIP) model as backbone along with Fast TCM-CR50 to achieve state of the art performance in inference speeds. It also achieved impressive performance in night-time scenes.

The text detectors discussed so far require extensive character-level annotations. Methods that attempt to alleviate this problem fail to recognize text that are completely curved or distorted geometrically. These models also fail if the text have uncommon fonts and styles. Many models try to solve one of these problems but none to the best of our knowledge solve all of these shortcomings.

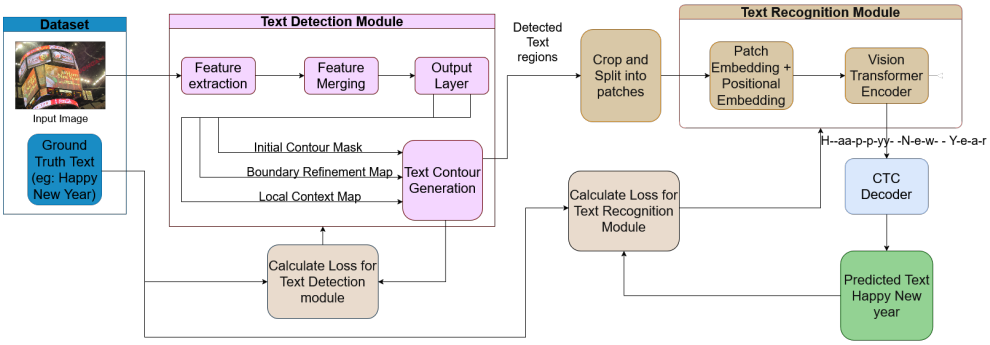


Fig. 1: System Diagram

3 Proposed Methodology

Our methodology predicts word only with the ground-truth text hence alleviating the requirement of extensive character-level annotations of existing methodologies. As our model trains to predict the relative position and character class in a word using visual features, it is able to recognize curved texts and texts in any orientation.

Our proposed scene text extractor consists of two modules: Text detection module and Text recognition module. The text detection module predicts a mask that predicts the locations of the detected texts. The detected text regions are cropped and passed through the text recognition module to extract the text in it.

3.1 Text detection module

Inspired by the work in [21], our detection branch is built to robustly locate text regions—even when texts appear rotated, distorted, or in unusual orientations. Our approach is divided into two main parts: 1) Initial contour mask generation, 2) Local Context Map (LCM)

1. Initial contour Mask Generation Fig.2 explains the architecture of the text detection network. The network produces a initial contour mask which is a mask image where each pixel represents the probability of the pixel belonging to a text, and a boundary refinement map in which each pixel value represents the shortest distance from that pixel to the text outline. Unlike [22] which used CNN and Ridgelet filters to extract low-level and high-level features our network uses the ResNet50 architecture as the feature extraction backbone. The feature maps from each stage of the ResNet is merged to get the combination of high level and low level features. The exact merging algorithm is as follows:

$$\mathbf{H}_i^1 = \text{conv}_{1\times1,128}(\mathbf{F}_i), \quad i = 1, 2, 3, 4, \tag{1}$$

$$\mathbf{H}_i^2 = \begin{cases} \mathbf{H}_i^1, & i = 1 \\ \text{upsample}_2(\mathbf{H}_i^1) + \mathbf{H}_{i-1}^2, & i = 2, 3, 4 \end{cases} \tag{2}$$

$$\mathbf{H}_i^3 = \begin{cases} \text{conv}_{3\times3,64}(\mathbf{H}_i^2), & i = 1 \\ \text{upsample}_{2^{(i-1)}}(\text{conv}_{3\times3,64}(\mathbf{H}_i^2), & i = 2,3,4 \end{cases} \tag{3}$$

$$\mathbf{F} = \text{concatenate}(\mathbf{H}_1^3, \mathbf{H}_2^3, \mathbf{H}_3^3, \mathbf{H}_4^3). \tag{4}$$

where F_i is the feature map of the ResNet backbone and H_i^1 ,H_i^2 and H_i^3 are hidden feature maps. F is the combined feature map that combines the low-level information and high-level semantics. Here *up* represents the upsampling operator.

The fused feature map H_f is passed on to two convolutional networks to produce the initial contour mask and local context maps. The exact architecture is given in the Fig. 2 output layer branch. The two networks produces outputs T_1 and T_2 respectively. T_1 is a tensor of size $\frac{H}{4} \times \frac{W}{4} \times 2$ and T_2 is a tensor of size $\frac{H}{4} \times \frac{W}{4} \times 9$. The T_1 consists of two tensors - predicted initial contour mask and the boundary refinement map T_r each of size $\frac{H}{4} \times \frac{W}{4} \times 1$.

The boundary refinement map is used to extend the initial contour mask with the offset calculated as following:

$$\text{offset} = \frac{T_r^{i,j} + T_r^{k,v}}{2} \tag{5}$$

2. Local Context Map LCM encourages the network to look beyond a single pixel by considering its surrounding neighbourhood. This additional context helps the model distinguish text pixels from similar background regions, especially when low-level features (like colour or texture) are ambiguous.In scenarios where the gap (or interval) between the initial contour mask and the actual text is subtle, nearby background regions might be misclassified as text. LCM addresses this by treating the interval regions as background. This helps to prevent "text adhesion," where adjacent text instances might merge together erroneously. It is however worth noting that LCM is a training only operator and is not used in inference. This increases inference speed as there is no extra computational cost. During training the LCM is generated as follows:

$$\text{LCM} = \frac{S_{(A_v \cap A_{1s})} + S_{(A_v \cap A_{2s})}}{S_{(A_v)}} \tag{6}$$

where A_{1s} and A_{2s} are initial contour mask regions.

The detected text contours was used to crop the image into multiple patches where each patch contain a single word of text. Each patch is passed through the text recognition branch explained below.

3.2 Text recognition module

Traditional segmentation techniques distinguish each and every pixel as text or non-text region to localize the position of characters in the image. This process requires large character-level annotations (coordinates of character bounding box in image) which are time consuming to collect. They also struggle with grouping the detected

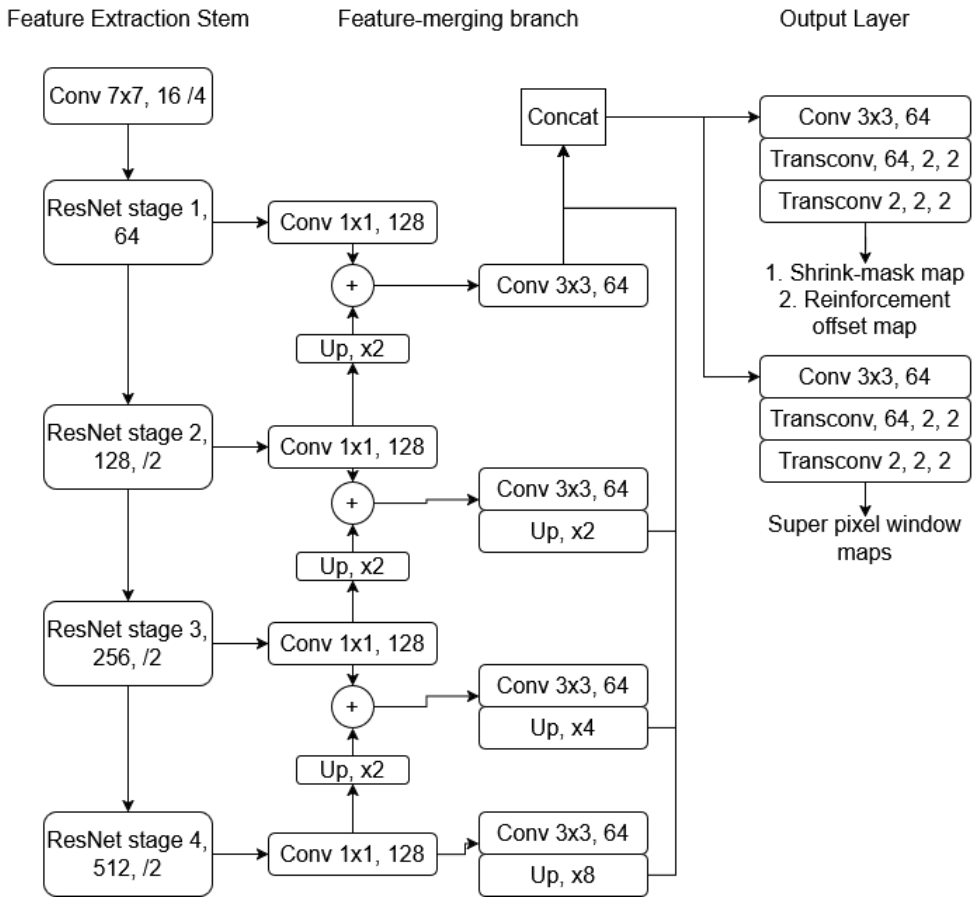


Fig. 2: Text detection network architecture

characters due complex geometry and alignment. Our proposed text recognition module based on vision transformer doesn't require extensive character level annotations. The network predicts a set of characters in the correct order and length.

3.2.1 Working of text recognition module

The vision transformer [2] was originally proposed for object recognition task. However our methodology predicts multiple characters in correct sequence and length. The vision transformer uses a mechanism called multi head self-attention which predicts the characters in parallel. The difference between vision transformer and standard transformer [1] is that the vision transformer only utilizes the encoder network. The original transformer was introduced for machine translation and other NLP tasks and it henceforth used word embeddings.

Instead of using word embeddings, the input image $x \in \mathbb{R}^{H \times W \times C}$ is reshaped into 2D patches which are then flattened into the size, $x^p \in \mathbb{R}^{N \times P^2 C}$. The transformer

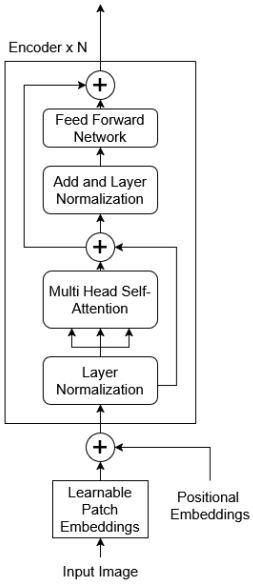


Fig. 3: Text recognition module architecture

takes these patches and processes them in parallel in different heads which makes them lose their positional information. Hence a positional encoding (PE) is added to the input of the encoder along with a learnable patch embedding. The encoder employs a normalization layer before each of the feed forward network.

The vision transformer network consists of multi-head self attention layers and feed forward networks. Attention layers are given by equation from [4]

$$Self - Attention(A, B, C) = \text{softmax}\left(\frac{AB^{\mathrm{T}}}{\sqrt{h}}\right)C \tag{7}$$

where A,B,C refer to query matrix, key matrix and value matrix respectively.h is dimension of A and B Multi Head attention layer is a concatenation of several single self attention heads. It linearly projects queries, keys, values M times. This is defined by

$$MultiHead(A, B, C) = Concatenate(head_1, ..., head_{\mathrm{M}})W^{\mathrm{o}}.$$

Where

$$head_{\mathrm{i}} = Self - Attention(AW_{\mathrm{i}}{}^{\mathrm{A}}, BW_{\mathrm{i}}{}^{\mathrm{B}}, CW_{\mathrm{i}}{}^{\mathrm{C}}) \tag{8}$$

Here $\mathrm{W_i}^{\mathrm{Q}}$, $\mathrm{W_i}^{\mathrm{K}}$, $\mathrm{W_i}^{\mathrm{V}}$ are learnt weights of queries, keys and values. The Feed Forward Network uses ReLU activation function for its hidden layers

$$ReLU(x) = max(0, x) \tag{9}$$

$$FeedForwardNetwork(x) = maximum(0, xW_1 + b_1)W_2 + b_2 \tag{10}$$

We use a [START] token instead of the learnable class embedding in the original vision transformer to mark the start of the text prediction. Another special token [PAD] is used to indicate the end of the text or white-space. The [PAD] token is appended to the end of each text prediction to match the maximum prediction length. Multiple vectors are extracted from the encoder, with the maximum length set to the longest text length in the dataset plus two additional tokens: one for [START] and one for [PAD].

3.3 Word Formation

The output of the vision transformer encoder is max_length number of vectors each of size D which corresponds to the embedding size. Each vector is mapped to a character by taking the maximum probability character. The detected characters may contain [PAD] tokens in between them or it may contain duplicate characters. These can be resolved by using a Connectionist Temporal Classification (CTC) Decoder [23]. The exact algorithm is given in algorithm 1

Algorithm 1: CTC Beam Search Decoding

Input: Probabilities P of shape $(T \times C)$, Beam Width B
Output: Best decoded sequence
Initialize Beam Set: $B_t = \{(\text{""}, 1.0)\}$;
for $t = 1$ *to* T **do**
  Initialize New Beam Set: $B_{t+1} = \{\}$;
  for *each* $(seq, prob)$ *in* B_t **do**
    for *each character* c *in possible characters* **do**
      $new\_seq = seq + c$;
      $new\_prob = prob \times P(t, c)$;
      if new_seq *exists in* B_{t+1} **then**
        $B_{t+1}[new\_seq] += new\_prob$;
      else
        $B_{t+1}[new\_seq] = new\_prob$;
      end
    end
  end
  Prune B_{t+1} to keep only top B sequences;
  Normalize probabilities;
  $B_T = B_{t+1}$;
end
return the highest probability sequence in B_T;

3.4 Training

The aim of the network is to minimize the following loss function

$$L = L_{\text{det}} + L_{\text{recog}} \tag{11}$$

where L_{det} and L_{recog} are the loss of text detection and text recognition tasks respectively.

3.4.1 Text Detection

The initial contour mask segmentation is optimized using the dice loss L_{icm}. The dice loss is based on the difference between binary segmentation masks and their ground truth labels.

$$L_{\text{icm}} = 1 - \frac{2 \cdot |P \cap G| + \epsilon}{|P| + |G| + \epsilon} \tag{12}$$

where P is the predicted binary mask, G is the ground truth, and ϵ is a small constant for numerical stability.

Ratio loss is used to optimize both boundary refinement map module and the Local Context Map module.

$$L_{ratio}(y, \widehat{y}) = log \frac{maximum(y, \widehat{y})}{minimum(y, \widehat{y})} \tag{13}$$

where y and $\widehat{y}$ are the ground truth and the prediction respectively. Hence the losses for boundary refinement map and the local context map are calculated as follows

$$L_{offset} = L_{ratio}(offset, \widehat{offset}) \tag{14}$$

$$L_{LCM} = L_{ratio}(LCM, \widehat{LCM}) \tag{15}$$

Hence the overall text detection loss is given by

$$L_{det} = \lambda_1 L_{icm} + \lambda_2 L_{offset} + \lambda_3 L_{LCM} \tag{16}$$

3.4.2 Text Recognition

The character prediction is a multi-class classification problem and hence cross-entropy loss is used.

$$L_{recog} = L_{CE}(P, \widehat{P}) \tag{17}$$

where L_{CE} is the standard cross-entropy loss, G is ground-truth text and P is the predicted text.

4 Experiments

In this section we present the dataset details, evaluation metrics, implementation results and experimental results and comparison against state-of-the-art models in sections 4.1, 4.2, 4.3, IV-D respectively.

4.1 Datasets

We train and evaluate our model on many widely used publicly available benchmark datasets. The list of datasets used along with number of training and testing images are listed in the Table 1. **SynthText** and **IIIT5K** are regarded as regular datasets. They contain images with mostly horizontal text with minimum distortion or rotation. **Total Text** and **SCUT-CTW1500** datasets are curved datasets where most texts are curved, distorted or rotated.

Table 1: Dataset Summary

Dataset	Number of Training Images	Number of Testing Images
SynthText	9M	-
ICDAR15	1,000	500
SCUT-CTW1500	1000	500
CUTE80	-	80

4.2 Evaluation Metrics

The performance of text extraction is evaluated using Levenshtein ratio. Levenshtein distance measures how dissimilar two words are by counting the smallest number of single-character operations—such as insertions, deletions, or substitutions—needed to transform one word into the other. When this distance is divided by the total length of the alignment, it yields the Levenshtein ratio. It essentially provides a similarity score between any two strings. Levenshtein distance is calculated as shown in equation 18.

$$\text{levenshtein}(x,y) = \begin{cases} i & \text{if } y = 0 \\ j & \text{if } x = 0 \\ \min \begin{cases} \text{levenshtein}(x-1,y)+1 \\ \text{levenshtein}(x,y-1)+1 \\ \text{levenshtein}(x-1,y-1)+\text{c} \end{cases} & \text{otherwise} \end{cases} \tag{18}$$

where cost $= 0$ if $s1[x] = s2[y]$, otherwise c $= 1$.

The Levenshtein ratio, which measures string similarity, is given by:

$$\text{Levenshtein Ratio} = \frac{\text{len}(s1) + \text{len}(s2) - \text{lev}(s1, s2)}{\text{len}(s1) + \text{len}(s2)} \tag{19}$$

To evaluate the text detection phase precision, recall and f1-score are used.

We evaluate the model using the test images provided in the datasets mentioned in Table 1

4.3 Implementation Details

The ResNet50 backbone pretrained on the ImageNet dataset has been utilized as the text detection module. We set the learning rate of this text detection branch as 0.001 and batch size as 16. The backbone learning rate is adjusted to 10^{-5}, whereas transformer is set at 10^{-4}. The batch size is 48 for text extraction branch and we have 12 transformer encoders for text detection module. In the encoder, each multi-head attention layer and every feed forward network (FFN) layer has a dropout layer with a dropout rate of 0.1. It is capable of classifying 63 characters, of which are digits (0 to 9), lowercase letters (a to z), uppercase letters (A to Z), [START] and [PAD] token. Sequence length is set to a max of 27, 25 for characters plus [START] and [PAD] tokens. To add the diversity to the training data and guide the model to learn to work with these variations, random cropping, random rotation, random scaling, random horizontal flipping etc. are used for data augmentation.

Fig. 4: Input Image

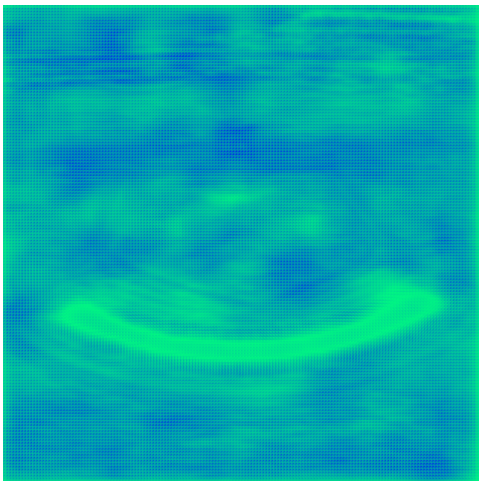
4.4 Results and Discussions

4.4.1 Text Detection

As mentioned in Table 6.1 three datasets have been used for testing. ICDAR15, SCUT-CTW1500 and CUTE80 are the three datasets. Images of size $640x640$ are passed through the text detection module. The results from the text detection modules are presented below for the sample input in figure 4. The sample image is from the CUTE80 dataset.

As depicted in the Figure 5 the text in the image is curved and our model predicted the initial contour mask accurately. All the characters are covered within the initial contour mask map. The initial contour mask contour is extended outward by the boundary refinement offset and the detected text region is visualized in Figure 6.

(a) Predicted Initial contour mask for the input image.



(b) Predicted boundary refinement map.

Fig. 5: Visualization of predicted outputs.

The initial contour mask was a tensor of size 160 x 160 x 1 (H/4* W/4*1) and the boundary refinement map was also a tensor of size 160 x 160 x 1 (H/4* W/4*1). The local context map was omitted in inference to improve inference speed.

Using the above equations the evaluation metrics for the text detection module has been calculated for all datasets and the results are presented in Table 2. The confusion matrix for each dataset is presented in Figures 8.

Table 2: Text Detection Metrics over multiple datasets

Dataset	Precision %	Recall %	F1 Score %
ICDAR15	86.42	66.70	75.29
SCUT-CTW1500	84.64	88.28	86.42

The results of the text detection module have been compared with other State Of The Art (SOTA) models for various datasets and the results are presented in Table 3

4.4.2 Text Recognition

The text extraction phase was done on the detected text regions cropped from the original image. The cropped image is depicted in Figure 7.

The text extraction result for the image was **"WEST HAM UNITED"**. The attention for the image is depicted in Figure 9.

Fig. 6: Detected text in the image

Fig. 7: Detected text is cropped from the image

Table 3: Text Recognition Metrics over multiple SOTA methods on SCUT-CTW1500

SOTA Methods	Precision %	Recall %	F1 Score %
MixNet [24]	91.4	88.3	89.8
SRFormer [25]	91.6	87.7	89.6
DPText-DETR [26]	91.7	86.2	88.8
TextFuseNet [27]	89.7	85.1	87.4
Ours	84.64	88.28	86.42

Using the equation 18 the evaluation metrics for the text detection module has been calculated for SCUT-CTW1500 dataset and ICDAR15 dataset and the results are presented in Table 4.

The experimental results have been presented in the above tables and figures. Texts that are in various rotation and orientation are accurately detected by our model, as is evident from table 6.6. However, some regions are also falsely detected as texts

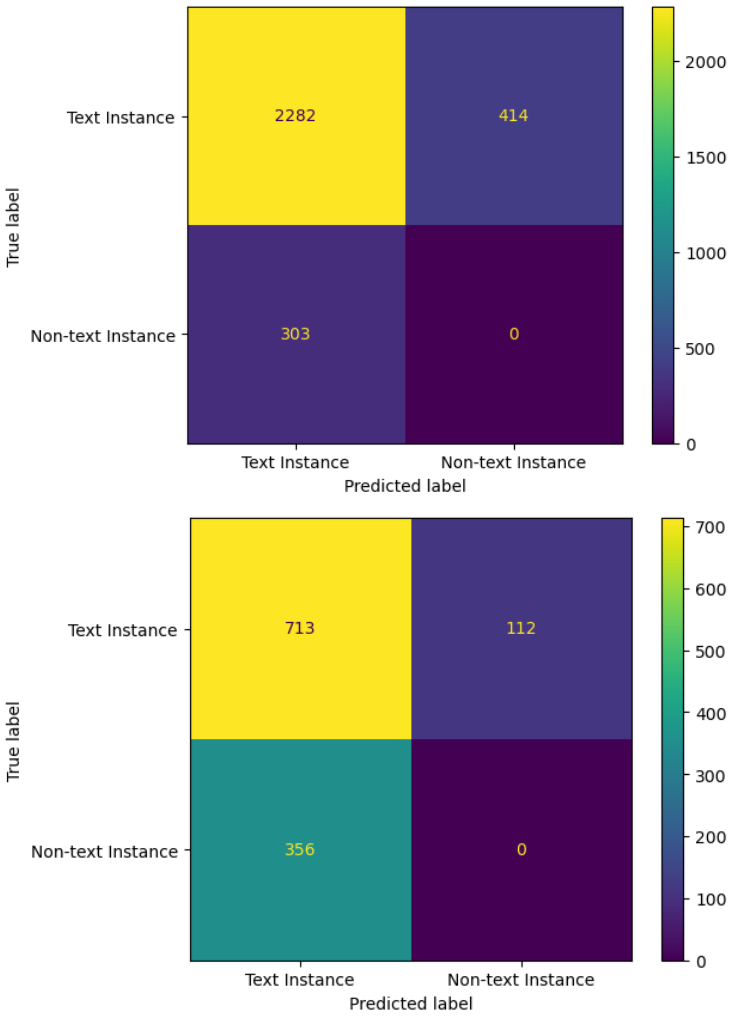


Fig. 8: Confusion matrices for SCUT-CTW1500 dataset, ICDAR15 dataset and CUTE80 dataset.

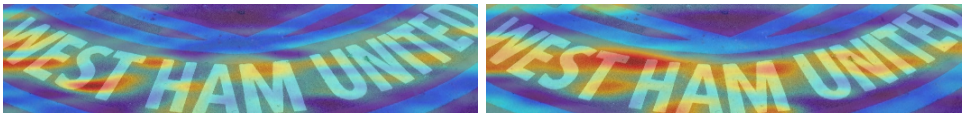


Fig. 9: Visualization of attention maps.

Table 4: Text Recognition Metrics over multiple datasets

Dataset	Levenshtein Ratio
ICDAR15	85.1
SCUT-CTW1500	84.6

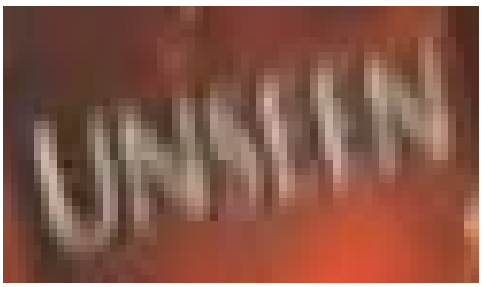
when, in fact, there is no text in that particular region in the image. Some examples are given in figure 10.

Fig. 10: Visualization of attention maps.

In very few images our model recognizes designs or patterns that resemble characters in shape or size as text regions. Our model extracts partial texts from very low resolution images like in figure 11

4.5 Future enhancements

While the approach of utilizing initial contour mask and vision transformer to recognize the text enhances the accuracy of the model, it has some limitations that require addressing in future work. One significant limitation is that the model is trained using only English characters and digits. Therefore, to expand the scope of the model and make it more versatile, it is necessary to increase the training dataset by including additional languages. By expanding the dataset, the model can be trained to identify more languages like Tamil, Chinese, etc. and special symbols like #, @, $, which would increase its usefulness and applicability in various scenarios. Additionally, the model has been found to underperform for images captured in low light conditions, such as during night time. Such scenarios could be explored further.



(a) Predicted:unle (Actual: unseen)

(b) Predicted: adid (Actual: adidas)

(c) Predicted: sde (Actual: sale)

(d) Predicted: arc (Actual: marc)

Fig. 11: Illustration of failure cases.

5 Conclusion

Enhancements in text detection methodologies are addressed by two works, though the problem is approached from different perspectives. In the first work a reinforcement learning-based method is bring forward to improve text localization by having find out regions progressively shrunk, thereby permitting false positives to be decreased and detection accuracy to be increased. In the second work, another aspect of text detection and recognition is explored, mostly focusing on the hard conditions involved in multi-oriented text or scene text recognition. When the models from both works are merged, it is made notable that better enhancement in text detection under hard environments can be obtained using new deep learning techniques, mainly those based on reinforcement learning and CNN architectures. Detection outcomes are additionally processed by the use of better methodologies such as the shrink-mask mechanism, makes them more strengthen towards noise and low-resolution images. By such techniques being combined, state-of-the-art performance can be gained in real time applications where text is aligned in arbitrary orientations and diverse backgrounds.

References

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł.: Attention is all you need. In: 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA (2017)

[2] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: ICLR 2021 (2021)

[3] Zhang, X., Zhang, Y.: Conv-pvt: a fusion architecture of convolution and pyramid vision transformer. International Journal of Machine Learning and Cybernetics **14**, 2127–2136 (2023) https://doi.org/10.1007/s13042-022-01750-0

[4] Shivakumara, P., Banerjee, A., Pal, U., Nandanwar, L., Liu, C.-L.: A new language-independent deep cnn for scene text detection and style transfer in social media images. IEEE Transactions on Image Processing **32** (2023)

[5] Ma, J., Guo, S., Zhang, L.: Text prior guided scene text image. IEEE Transactions on Image Processing **32** (2023)

[6] Zhong, D., Zhan, H., Lyu, S., Liu, C., Yin, B., Shivakumara, P., Pal, U., Lu, Y.: Ndorder: Exploring a novel decoding order for scene text recognition. Expert Systems With Applications **249** (2024)

[7] Banerjee, A., Palaiahnakote, S., Pal, U., Antonacopoulos, A., Lu, T., Canet, J.L.: Tts: Hilbert transform-based generative adversarial network for tattoo and scene text spotting. Engineering Applications of Artificial Intelligence **133** (2024)

[8] Rong, X., Yi, C., Tian, Y.: Unambiguous text localization, retrieval, and recognition for cluttered scenes. IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(3) (2022)

[9] Wu, L., Xu, Y., Hou, J., Chen, C.L.P., Liu, C.-L.: A two-level rectification attention network for scene text recognition. IEEE Transactions on Multimedia **25** (2023)

[10] Liu, Y., Shen, C., Jin, L., He, T., Chen, P., Liu, C., Chen, H.: Abcnet v2: Adaptive bezier-curve network for real-time end-to-end text spotting. IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(11) (2022)

[11] Cao, M., Zhang, C., Yang, D., Zou, Y.: All you need is a second look: Towards arbitrary-shaped text detection. IEEE Transactions on Circuits and Systems for Video Technology **32**(2) (2022)

[12] Bi, H., Xu, C., Shi, C., Liu, G., Zhang, H., Li, Y., Dong, J.: Hgr-net: Hierarchical graph reasoning network for arbitrary shape scene text detection. IEEE Transactions on Image Processing **32** (2023)

[13] Yang, C., Chen, M., Xiong, Z., Yuan, Y., Wang, Q.: Cm-net: Concentric mask based arbitrary-shaped text detection. IEEE Transactions on Image Processing **31** (2022)

[14] Wang, W., Xie, E., Li, X., Liu, X., Liang, D., Yang, Z., Lu, T., Shen, C.: Pan++: Towards efficient and accurate end-to-end spotting of arbitrarily-shaped text. IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(9) (2022)

[15] Wang, Q., Fu, B., Li, M., He, J., Peng, X., Qiao, Y.: Region-aware arbitrary-shaped text detection with progressive fusion. IEEE Transactions on Multimedia **25** (2023)

[16] Moitra, M., Saha, S.K.: A review on handwritten text segmentation in indian languages. International Journal of Machine Learning and Cybernetics (2024) https://doi.org/10.1007/s13042-024-02448-1

[17] P, G.J., B, A., T, M., M, M.U.: Scene text detection and recognition using maximally stable extremal region. Journal of Applied Engineering and Technological Science (JAETS) **6**(1), 103–114 (2024) https://doi.org/10.37385/jaets.v6i1.5958

[18] Zhou, Y., Xie, H., Fang, S., Zhang, Y.: Semi-supervised text detection with accurate pseudo-labels. IEEE Signal Processing Letters **29** (2022)

[19] Wang, P., Li, H., Shen, C.: Towards end-to-end text spotting in natural scenes. IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(10) (2022)

[20] Yu, W., Liu, Y., Zhu, X., Cao, H., Sun, X., Bai, X.: Turning a clip model into a scene text spotter. IEEE Transactions on Pattern Analysis and Machine Intelligence **46**(9) (2024)

[21] Zhang, S.-X., Yang, C., Zhu, X., Zhou, H., Wang, H., Yin, X.-C.: Inverse-like antagonistic scene text spotting via reading-order estimation and dynamic sampling. IEEE Transactions on Image Processing **33** (2024)

[22] Nisia, T.G., Rajesh, S.: Extraction of high-level and low-level feature for classification of image using ridgelet and cnn based image classification. Journal of Physics: Conference Series **1911**(1), 012019 (2021) https://doi.org/10.1088/1742-6596/1911/1/012019

[23] Graves, A., Fernández, S., Gomez, F., Schmidhuber, J.: Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In: Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA (2006)

[24] Zeng, Y.-X., Hsieh, J.-W., Li, X., Chang, M.-C.: MixNet: Toward Accurate Detection of Challenging Scene Text in the Wild. arXiv:2308.12817 (2023)

[25] Bu, Q., Park, S., Khang, M., Cheng, Y.: SRFormer: Text Detection Transformer with Incorporated Segmentation and Regression. arXiv:2308.10531 (2023)

[26] Ye, M., Zhang, J., Zhao, S., Liu, J., Du, B., Tao, D.: DPText-DETR:

Towards Better Scene Text Detection with Dynamic Points in Transformer. arXiv:2207.04491v2 (2022)

[27] Ye, J., Chen, Z., Liu, J., Du, B.: Textfusenet: Scene text detection with richer fused features. In: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20) (2020)