High Precision Text Line Segmentation in Palm Leaf Manuscripts

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science in Electronics and Communication Engineering by Research

by

Niharika Vadlamudi
2018122008
niharika.vadlamudi@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
March 2025

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "High Precision Text Line Segmentation in Palm Leaf Manuscripts" by Niharika Vadlamudi, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Advisor: Dr. Ravi Kiran S

To

My family,

My teachers and

My friends

Acknowledgments

Firstly, I extend my profound gratitude to IIIT Hyderabad for their exemplary Lateral Entry (LE) program, which turned out to be a turning point in my life. I am grateful to all IIIT staff , assistants , professors , particularly Dr. Ravi Kiran S for being an incredibly supportive research guide in all my highs and lows . Under his guidance , I was able understand the essence of 'research' and tune my engineering and soft skills.I extend my gratitude to the HDIA team-Sowmya Aitha, Abhishek Trivedi, Waseem H Muhammad, Vaibhav Agarwal, Sreenya Chitluri, Amal Joseph for the great partnership. I specially would like to thank Rahul Krishna for being the best project partner , along with B.V.Khadiravarna and Naren Akash for joyful moments and great conversations during hard times.

I would like to thank my mother, Macharla Padmaja, for her constant support and care throughout my entire IIIT journey. As one of my friends put it best, all of my successes translates to her success. My sweetest brother, Prasanth V.V, for all the letting me experience his love and warmth in this lifetime. Thanks to my father, V.N.M Rao, for keeping things light when they got out of control and for caring for my health after every sleepless night.

To my special set of friends I found at IIIT who made the journey filled with memories , laughter and warmth. To Sivani Pentapati, an irreplaceable best friend for being my rock and constantly motivating me through obstacles, and always cheering me in my moments of happiness and success. Thejasvi Konduru and Harshithanjani Athi, for filling our cold 11 pm walks filled with hot gossips. Deeksha Devendra for being the best roommate ever, Anurag Sahu for taking care of me, and Rajashekhar Reddy for all the crazy fun memories. Thank you for dragging me into all your adventures and always loving me in your own little ways. Finally, to Kadiyala Abhiram, whose insightful conversations on politics, feminism, HIMYM, and every other topic under the sky will be cherished forever.

To my MEC friends and family , I'm forever indebted to Praharsha Chilamakuri (Sanjay) for always being a phone call away for pacifying me and being my constant source of positive affirmation. I'd like to thank my extremely rewarding female friendships with Steffi Roy, Kavya Narukurti, Mounika Vakulabharanam, and Bhavika Krishnam. I extend my gratitude to my former professors at MEC - P BhanuKiran, K.R Sarma, Vijayshekhar Chellaboina, J.L Bhattacharya, Mahipal Jetta, Jai Prakash, Kaushik Basak, Gomathi A, Naga D, Nagaih Chamakuri, Chithra Gurnani, Tabitha C, Salome Benhur, and Bhargava Rajaram for their instrumental role in shaping me into a better student and for believing in my potential to excel academically. To all those mentioned above and to anyone else who has played a part in shaping my journey, thank you from the bottom of my heart for all love and support.

Abstract

Ancient manuscripts were among the first forms of written communication, offering key insights into our past, covering literature, medicine, culture, philosophy, religion, and more. It is imperative to save these writings to identify and extract their hidden knowledge. Document collections frequently exhibit overlapping components, irregular patterns, dense layouts, extremely high aspect ratios, physical and chemical degradation (evident in ink-based manuscripts), text misalignment, and more. Compounding these difficulties are issues that may arise during the digitization processes, such as improper document positioning, inadequate illumination, and scanner effects. In this academic thesis, our main emphasis is on identifying and segmenting text lines inside these documents for downstream OCR applications with utmost precision.

We devise a two-stage approach - SeamFormer- to identify special text baselines in palm-leaf manuscripts using a multi-task strategy via Vision Transformers (ViTs). In the first stage, we detect text strikethroughs, namely 'scribbles,' which act as pointers to the location of text line segments within the document. The encoder-decoder architecture is used to analyze input image patches and produce two separate maps: a scribble map and a binary map. In the second stage, we post-process the prior stage outputs and generate a diacritic-aware global energy map. To generate the final precise text line polygons, we use a modified seam generation algorithm along with customized global maps.

The prior methodology is further enhanced by the proposed LineTR model, a multi-task DETR (Detection Transformer) model that reconceptualizes the scribble generation as a geometric problem. This method simply generates line parameters for each text line present in the input image patch. This design decision enables the model to exhibit zero-shot behavior across diverse historical manuscripts. The state-of-the-art approach has been proven to generate precise text-line segmentation with a single 'unified' model with minimal post-processing efforts, making it a strong candidate for image-to-OCR integration pipelines.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Ancient manuscripts were among the first forms of written communication, offering key insights into our history. As a society, preserving our rich heritage and identifying and extracting the hidden knowledge of these manuscripts is imperative. A comprehensive OCR pipeline involving various entities such as language specialists, annotation experts, annotation software systems, task-specific architectures, etc, is required to extract knowledge from manuscripts effectively. A typical Optical Character Recognition (OCR) pipeline involves several stages, as shown in Fig. 1.1. A text line detector is leveraged to extract text lines in the document image that are further processed by language-specific OCR systems to extract text.

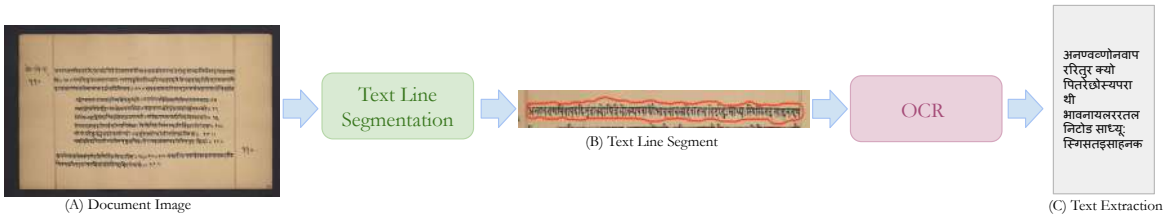


Figure 1.1: An end-to-end image-to-text pipeline integrates a document text line segmentation module followed by OCR to extract text from the segmented lines.

To develop effective image-to-text OCR pipelines, it is important to address the problem of text line segmentation within the target document collections . This primary step is crucial, as the accuracy of OCR transcription is influenced by the precision of text line segmentation. Inaccurate segmentation, particularly the omission of diacritics within text lines, can lead to significant misinterpretations of the text. We demonstrate this by taking an example of Indic historical manuscript, which has numerous diacritical elements as part of the language vocabulary (refer Fig. 1.2).

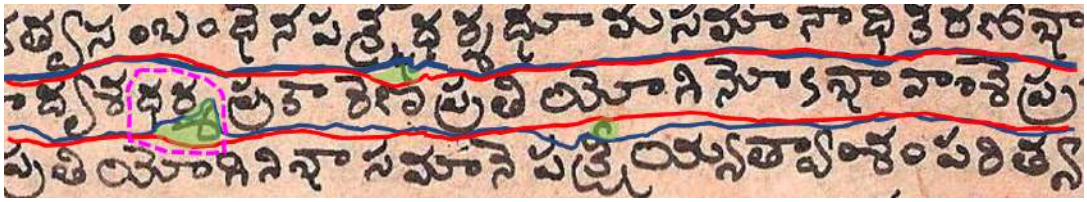


Figure 1.2: An example to illustrate the importance of precise text line segmentation in historical manuscripts. The ground truth upper and lower portions of the enclosing line annotation are shown in red. The prediction is shown in blue. The green shaded portions indicate crucial text fragments omitted by prediction causing the semantic interpretation of text to change. For e.g., pink dashed region encloses a word from the Indic language Telugu which means '**moral duty**'. The incorrect boundary prediction causes the resulting line to contain a word with a drastically different meaning '**price**'.

1.2 Challenges for Text Line Segmentation

Researchers often model the problem of text line annotation in historical handwriting as a semantic or instance segmentation problem. A substantial portion of the recent literature demonstrates the adaptation of generic computer vision architectures [6], [7] for application to the domain of documents, encompassing both structured and unstructured formats [1], [2]. However, unlike conventional vision datasets, documents present an additional layer of complexity due to the presence of text. Crucial information is often lost when techniques like image resizing are adopted, as they condense text pixels into indistinct pixelated blobs, devoid of the shape or curvature necessary to map them into definitive characters components (refer Fig. 1.3).

(A) Input Image (1166,3206,3) (B)Resized Image (256,256,3)

Figure 1.3: This image presents an ASR manuscript (A) from an Indic collection is resized to typical input dimension of (256,256) exhibiting significant degradation in text characters and a reduction in interline spacing (B). These issues complicate the accurate delineation of text lines.

Unlike regular document datasets, palm leaf manuscripts contain non-textual elements such as pictures, tables, and document decorators. These elements are frequently interspersed with text in non-standard layouts. For example, palm leaf manuscripts often contain one or more physical holes designed for stacking and binding the leaves together as a book. These holes, situated in the document's center,

disrupt the continuity of lines, thereby complicating the task of text line association for segmentation models (refer Fig. 1.4).

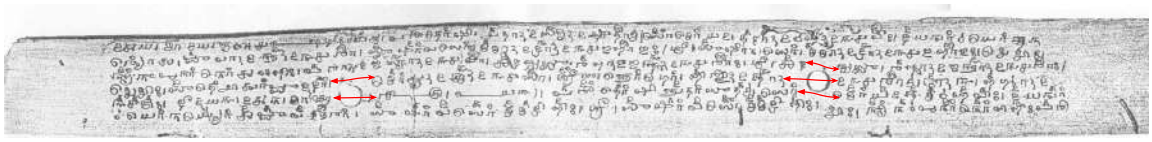


Figure 1.4: A Bhoomi Collection manuscript with red arrows highlighting shift in text line association around physical holes.

Another significant difference between traditional computer vision datasets (COCO [8], PASCAL VOC [9]) and text-line datasets is the contrast in ground truth annotations. To understand better, a typical image annotation of objects such as a bus or a person has a definitive boundary. However, a text line, as depicted in the Fig. 1.5, can have varying annotations depending on the annotator. Other issues in text-line annotations include overlapping instances across ground truth. Given that there are multiple sources of datasets, with no global standard procedure of polygonal annotation, results in varying precision of ground truth, thus impacting the training and evaluation of text-line segmentation models on large-scale pooled datasets.

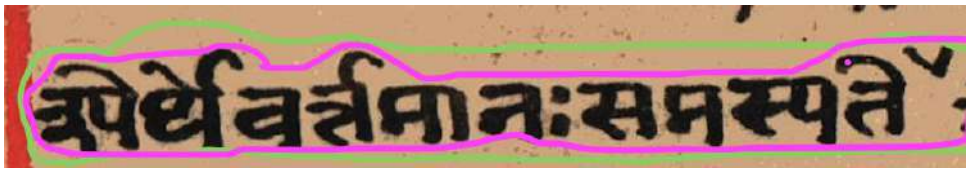


Figure 1.5: A sample image shows text line annotations in pink and green, which are considered valid ground truth polygons. However, the green annotation is loosely fit, affecting both training and evaluation when classic semantic segmentation metrics like IoU are applied.

Additionally , these handwritten nature of these documents and the surface material result in extremely uneven and dense lines , sometimes with varying densities within a document (-intra) and across pooled palm leaf datasets (intra) making it difficult to have a single shot model.(refer Fig. 1.6) These historical manuscripts, especially the palm-leaf ones, tend to be inherently fragile and prone to damage due to various sources wood-and-leaf-boring insects, humidity seepage, improper storage and handling etc. While some degradations cause the edges of the document to become frayed, others manifest as irregularly shaped perforations in the document interior. It may be important to identify such degradations before attempting lexically-focused tasks such as OCR or word-spotting or segmentation (refer Fig. 1.6).

As part of this thesis work , we aim to resolve few of these domain specific issues by adopting multi-stage approaches and shifting away from direct semantic segmentation techniques. To tackle multiple diverse datasets in a single shot manner , we adopt resizing and multi-scale context aware techniques.

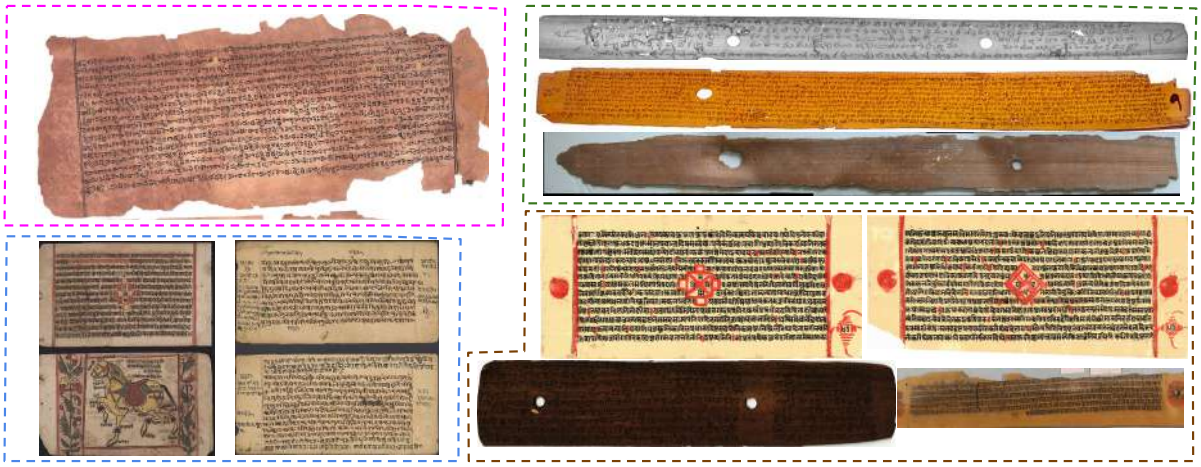


Figure 1.6: Representative manuscript images from Indiscapes2 dataset [1] from ASR collection (top left, pink dotted line) depicting text skew , Penn-in-Hand (bottom left, blue dotted line) showcasing multiple non-textual elements (pictures) and Bhoomi collection (green dotted line) with high aspect ration and text density.

1.3 Thesis Contribution

In this thesis, we take a significant step to address the above challenges by proposing end-to-end pipeline from image to text line polygon.

1.3.1 SeamFormer

We devise a two-stage approach - SeamFormer, for identifying special baselines in palm leaf manuscripts using a multi-task strategy via Vision Transformers (ViTs). In the first stage, we detect text strikethroughs, namely 'scribbles,' which act as pointers to the location of text lines segments within the document. The encoder-decoder architecture is used to analyze input image patches and produce two separate maps: a scribble map and a binary map. In the second stage , we post-process the prior stage outputs and generate diacritic aware global energy map.A modified seam generation algorithm, along with the customized global maps, is used for generating the final precise text line polygons.

1.3.2 LineTR

The prior methodology is further enhanced by the proposed LineTR model, a multi-task DETR (Detection Transformer) model that reconceptualizes the scribble generation as a geometric problem. In this method, it simply generates line parameters for each text-line present in the input image patch. This design decision enables the model to exhibits zero-shot behaviour across diverse historical manuscripts . The state-of-the-art approach has been proved to generate precise text-line segmentations with a single "unified" model with minimal post-processing efforts , making it a strong candidate for image-to-OCR integration pipelines.

1.4 Thesis Layout

The compilation of the thesis is as follows :

Chapter 1 This is the introductory chapter , which provides insights into the domain of historical handwritten documents.

Chapter 2 discusses the existing approaches dedicated for text-line segmentation.

Chapter 3 introduces SeamFormer, a novel resizing free approach for high precision fully automatic text-line segmentation. We provide details about the network architecture and training approach with qualitative and quantitative approach.

Chapter 4 highlights LineTR , a zero-shot DETR based unified model that revamps SeamFormer model and exhibits state-of-the-art performance accross multiple unseen challenging manuscripts.

Chapter 5 concludes the thesis by consolidating its contributions and proposing potential future solutions that can be implemented by researchers interested in further exploring this domain.

Chapter 2

Existing Works for Text Line Segmentation

In this chapter , we highlight recent text-line segmentation solutions in the domain of historical handwritten documents. As discussed in the previous chapter , handwritten documents require more fine-grained approach for tackling multiple constraints. Some of these solutions are often extended to further extended to segmentation of other non-textual elements. We begin with a concise overview of existing line segmentation studies, followed by an examination of networks designed specifically for palm leaf manuscripts pertaining South & South East Asian documents.

2.1 General Overview

Many approaches have been proposed for text line segmentation in other (i.e. non palm leaf) historical documents. To encourage research, many historical document datasets with line segmentation annotations have been introduced and utilized in competitions at premier document analysis venues. The categorisation of these research papers can be done on the basis of their approach top-down/bottom-up, semi/fully automatic analysis, deep learning/classical image processing/hybrid approaches,single stage/multi-stage pipelines, etc.

2.1.1 Classical Image Processing Approaches

Early approaches favoured the use of classical digital image processing techniques followed by post processing. Alaei et al. [10] employ a painting technique for foregrounding smearing to tackle unconstrained handwritten text line segmentation for diverse languages. Grouping techniques utilizing nearest neighbor [11], learning algorithms [12], and heuristic rules [13] have also been employed for text line segmentation. Projection profiles are another popular top-down approach to isolate text lines [14, 15, 16, 17, 18]. However, profile-based approaches cannot cope with highly curved lines and uneven layouts. Adaptive Local Connectivity Map (ALCM) [19, 20] is another technique for localizing and extracting text lines directly from gray-scale images. Generally, these approaches employ handcrafted processing elements with hyperparameters which do not generalize well across multiple datasets. The methods tend to require dataset specific techniques for isolating text line elements (e.g. strokes, diacritics) and often fail to disentangle touching components across consecutive text lines – a

common occurrence in handwritten documents. In a survey paper, Kesiman et al. [21] consider palm-leaf manuscripts from South-East Asia and evaluate numerous line segmentation approaches developed for other (non-Asian) historical documents. Chamchong and Fung propose an adaptive partial projection (APP) technique [22], an improvement over their earlier partial projection approach [23] for line extraction in Thai manuscripts. Valy et al. [24] propose an approach which also employs connected components and projection profiles to determine medial positions of text lines followed by a path finding approach to mark the text line boundaries in Khmer manuscripts. Kesiman et al. [25] employ a similar approach for Balinese manuscripts. Apart from the assumption of a component-based script, these approaches inherit the shortcomings of projection-based works mentioned previously. Seam generation, an approach involving optimization over image-derived energy maps [26], is a popular approach for text line segmentation. Saabni and El-Sana introduce a seam generation algorithm based on an energy map calculated using Signed Distance Transform (SDT) for Arabic manuscripts [27]. However, the approach involves repeated energy map computations for each line and significant dataset-specific post-processing to tackle overlapping components and diacritics. Asi et al. [28] improve upon the aforementioned approach by replacing SDT with a geodesic distance transform energy map. This method fails to tackle elongated letters and widely separated diacritics. Nikolaos et al. [29] use a medial line obtained using a projection profile approach to guide seam generation for line segmentation in multiple historical datasets. However, the method requires dataset specific parameter tuning for various pipeline stages. In existing approaches [27, 28, 29, 30, 31], seam generation is generally used to separate text lines rather than *segment* them. As a result, extraneous isolated character fragments and noisy background elements present beyond the line's text content are often included as part of the line.

2.1.2 Deep Learning Approaches

In recent years, a number of deep learning based approaches have been employed as well [32, 33, 34, 35, 36, 37, 38, 31]. Most of these approaches use a variant of the popular U-Net [39] architecture. These methods have the appeal of being optimized end-to-end and work well on Western historical manuscripts. However, the approaches require drastic downsampling of input image which eliminates crucial inter-line information. Coupled with the boundary smoothing that occurs when the network predictions are upsampled, this leads to imprecise and unsatisfactory line segment boundary predictions for other types of historical manuscripts such as ours (i.e. palm leaf). Jindal and Ghosh [40] use a Faster-RCNN model to obtain bounding boxes for a collection of Indic palm leaf manuscripts. However, this approach cannot tackle the curvature of lines which is present in almost all manuscripts. Prusty et al. [41] and Sharan et al. [1] propose approaches which modify the Mask-RCNN [6] framework for segmenting various semantic regions including text lines in Indic manuscripts. Despite their relatively better performance and ability to tackle line curvature, these approaches produce overly smoothed line boundaries and even tend to have false negatives (i.e. missed lines) on some occasions.

2.1.3 Hybrid Approaches

Alberti et al. [30] first employ a deep network to obtain a binarized version of the image. Seam generation is applied on the binary image to obtain coarse region boundaries, followed by a graph-based connected component procedure to obtain the polygonal line boundaries. The approach is not suitable for highly skewed and unevenly curving text found in palm leaf manuscripts. For enhancing the seam generation process, Nguyen et al. [42] introduce an additional global cost function for better detection of ascenders, descenders and diacritics. The approach is not suitable for skewed or curved text and requires heavy dataset-specific parameter tuning for the cost functions.

Given the above network architectures, we shift our focus to prior works that have been designed for Indiscapes2 [1] dataset.Most of these networks are designed to tackle high-aspect ratios , extreme degradation and heavy diacritical aspect of the dataset. These model architectures were designed to tackle all regions within a document image such as Character components , Library Marker, Physical Holes, Boundary Lines, Physical Degradation, Character Line Segments , Virtual Holes , etc. We will critically analyse only for Character Line Segment class.

2.2 Indiscapes: Instance Segmentation Networks for Layout Parsing of Historical Indic Manuscripts

This work [2] , leverages a variant of Mask RCNN model [6] a state-of-the-art model for popular datasets such as COCO , was further adapted to for document layout analysis in Indiscapes dataset. The network architecture contained three main stages :

- ***Backbone:*** The core of MaskRCNN pipeline was a ResNet-50 backbone, out of which only first 4 convolutional blocks were leveraged. This was used to extract features from the input image , and it consisted of convolutional network combined with feature pyramid network to enable multi-scale feature extraction.

- ***Region Proposal Network (RPN) :*** The primary objective of the Region Proposal Network (RPN) is to analyze multi-scale features within an image and produce prospective rectangular regions, referred to as "object proposals," denoting areas where objects of interest are likely to exist. Functioning as a basic convolutional network, the RPN is structured as a series of convolutional layers. Each of these rectangular proposals are generated , in the fixed aspect ratio for every multi-scale image. Within the RPN module, the anchor aspect ratios of 1:1,1:3,1:10 were chosen keeping the peculiar aspect ratios of manuscript images in mind and the number of proposals from RPN were reduced to 512. Along with this region proposal,we see i) probability score or 'objectness score' along with ii) offset coordinates of the bounding box with respect to the anchor. There is a filtering mechanism , where region proposals with maximum objectness score are retained, and

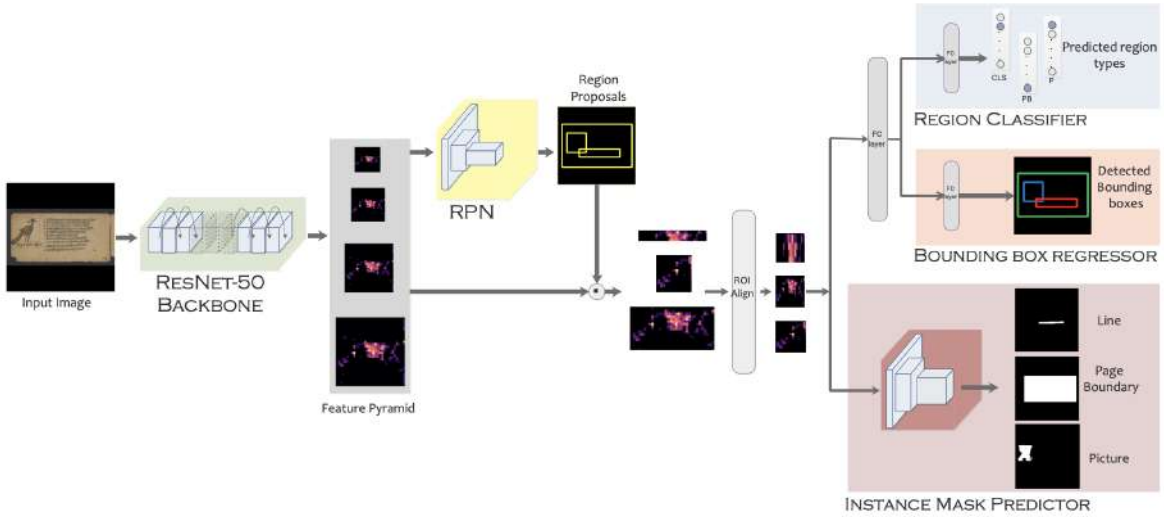


Figure 2.1: Mask RCNN Architecture

multiple overlapping boxes are removed via non-maximal supression. The final set of rectangular boxes are known as Region of Interest (RoI).

- ***Multi-Task Network Branches:*** The RoIs finalised from the RPN network are further warped into fixed dimensions and overlaid on feature maps extracted from backbone to obtain RoI-specific features. The combined features are propagated to 3 seperate parallel sub-networks namely : Region Classifier, Bounding Box Regressor & Instance Mask Prediction. The region classifier is fully-connected layer network which maps each of these feature regions to configured region labels(eg. Hole,Library Marker,Boundary Line) etc. The bounding box regressor maps the RoI features to bounding boxes. Instance segmentation branch is fully convolutional and predicts the pixel mask for the underlying region.

2.2.1 Results

Despite encountering challenges of Indic palm leaf manuscripts, the model achieves moderate performance across all classes. Particularly emphasizing the critical class of Character Line Segments, it effectively manages the overlap between physical Holes and Character Line Segments. However, the utilization of a resizing technique leads to the emergence of imprecise boundaries, which is reiterated in Fig. 2.3 Notably, within the Mask head, the Instance binary mask predicts a 28 x 28 canvas, a contrast to the average aspect ratio of a text-line, typically 1:20. Furthermore, the polygonal representations exhibit shortcomings, such as missing diacritics within dominant regions, a drawback with negative implications for downstream applications like Optical Character Recognition (OCR).

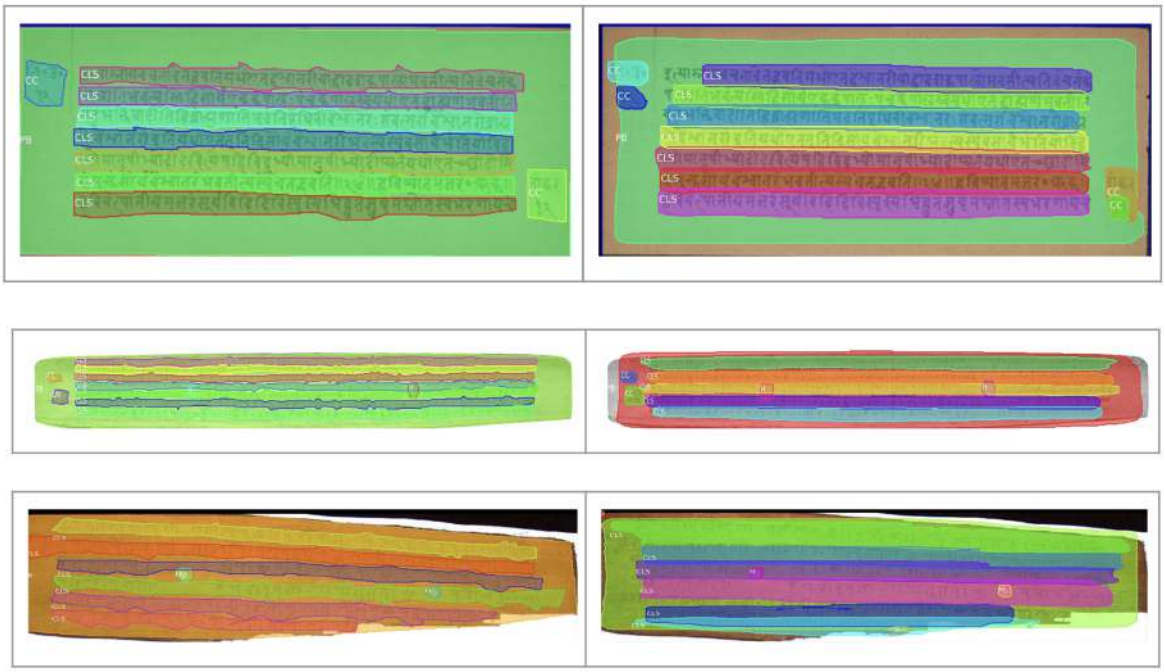


Figure 2.2: Here are two sample test images from the IndiscapesV1 dataset [2]– Bhoomi & Penn-In-Hand collection. Note that Ground Truth Annotations(left) and predicted instance segmentation(right).These outputs display instance masks with the **'CLS'** class. In these instances, polygon masks are generated for the text line instances, but they are often of mediocre quality. They frequently miss out on 'maatras' (diacritics), omit masks in certain regions, and are unable to handle orientation effectively.

Figure 2.3: For a single character line segment , we showcase the common problems presented by MaskRCNN [2] model.

2.3 PALMIRA: A Deep Deformable Network for Instance Segmentation of Dense and Uneven Layouts in Handwritten Manuscripts

The two main contribution of this work are i) advanced improvements from Prusty et al. [2] and introduction of a large scale Indic historical handwritten dataset known as Indiscapes2. To the proposed architecture , the two main modifications are :

2.3.1 Modification 1 : Deformable Convolutions in Backbone

Before examining the more general setting, let us temporarily consider 2D input feature maps x. Denote the 2D filter operating on this feature map as w and the convolution grid operating on the feature map as $\mathcal{R}$. As an example, for a 3×3 filter, we have:

$$\mathcal{R} = \begin{Bmatrix} (-1,-1) & (-1,0) & (-1,1) \\ (0,-1) & (0,0) & (0,1) \\ (1,-1) & (1,0) & (1,1) \end{Bmatrix} \tag{2.1}$$

Let the output feature map resulting from the convolution be y. For each pixel location p_0, we have:

$$y(p_0) = \sum_{p_n \in \mathcal{R}} w(p_n) \cdot x(p_0 + p_n) \tag{2.2}$$

where n indexes the spatial grid locations associated with $\mathcal{R}$. The default convolution operation in Mask R-CNN operates via a fixed 2D spatial integer grid as described above. However, this setup does not enable the grid to deform based on the input feature map, reducing the ability to better model the high inter/intra-region deformations and the features they induce.

As an alternative, Deformable Convolutions [43] provide a way to determine suitable local 2D offsets for the default spatial sampling locations (see Fig. 2.5a). Importantly, these offsets $\{\Delta p_n; n = 1, 2 \ldots\}$ are adaptively computed as a function of the input features for each reference location p_0. Equation 2.2 becomes:

$$y(p_0) = \sum_{p_n \in \mathcal{R}} w(p_n) \cdot x(p_0 + p_n + \Delta p_n) \tag{2.3}$$

Since the offsets Δp_n may be fractional, the sampled values for these locations are generated using bilinear interpolation. This also preserves the differentiability of the filters because the offset gradients are learnt via backpropagation through the bilinear transform. Due to the adaptive sampling of spatial locations, broader and generalized receptive fields are induced in the network. Note that the overall optimization involves jointly learning both the regular filter weights and weights for a small additional set of filters which operate on input to generate the offsets for input feature locations (Fig. 2.5a).

2.3.2 Modification 2 : Deforming the spatial grid in Mask Head

The 'Mask Head' in Vanilla Mask-RCNN takes aligned feature maps for each plausible region instance as input and outputs a binary mask corresponding to the predicted document region. In this process, the output is obtained relative to a 28×28 regular spatial grid representing the entire document image. The output is upsampled to the original document image dimensions to obtain the final region mask. As with the convolution operation discussed in the previous section, performing upsampling relative to a uniform (integer) grid leads to poorly estimated spatial boundaries for document regions, especially for our challenging manuscript scenario.

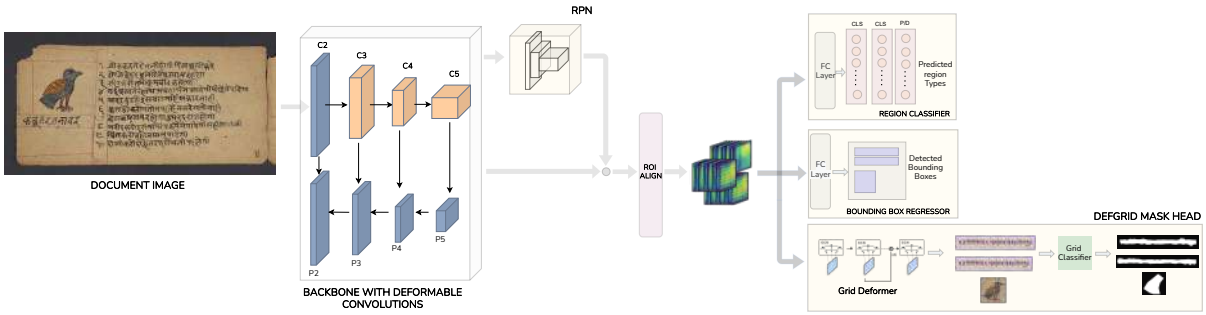
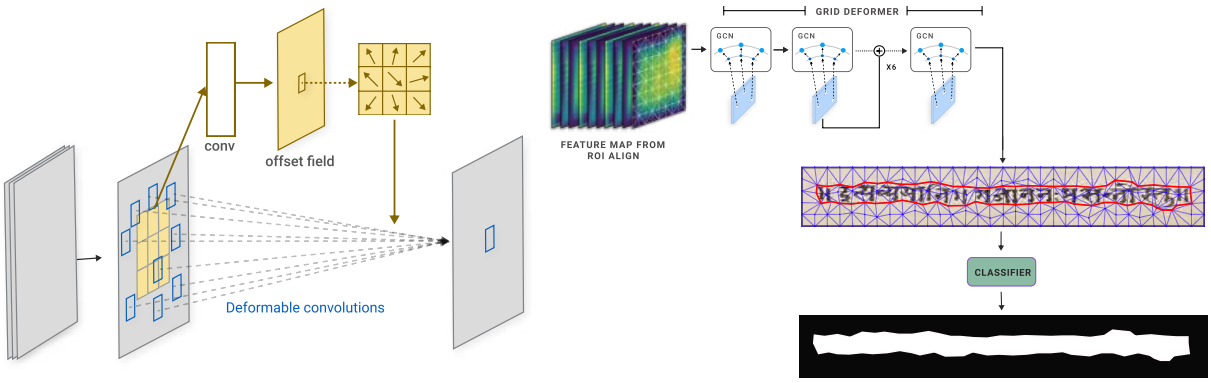


Figure 2.4: A diagram illustrating PALMIRA's architecture.The orange blocks in the back- bone are deformable convolutions and inclusion of Deformable Grid Mask Head which improved outputs region instance masks.



(a) Deformable convolution(Sec. 2.3).

(b) Deformable Grid Mask Head (Sec. 2.3.2).

Figure 2.5: Our novel modifications to the Vanilla Mask-RCNN framework (Sec. 2.3).

Similar in spirit to deformable convolutions, we adopt an approach wherein the output region boundary is obtained relative to a deformed grid. (see Fig. 2.5b). Let $F \in \mathbb{R}^{256\times14\times14}$ be feature map being fed as input to the Mask Head. Denote each of the integer grid vertices that tile the 14×14 spatial dimension as $v_i = [x_i, y_i]^T$. Each grid vertex is maximally connected to its 8-nearest neighbors to obtain a grid with triangle edges (see 'Feature Map from ROI Align' in Fig. 2.5b). The Deformable Grid Mask Head network is optimized to predict the offsets of the grid vertices such that a subset of edges incident on the vertices form a closed contour which aligns with the region boundary. To effectively model the chain-graph structure of the region boundary, the Mask Head utilizes six cascaded Residual Graph Convolutional Network blocks for prediction of offsets. The final layer predicts binary labels relative to the deformed grid structure formed by the offset vertices (i.e. $v_i + [\Delta x_i, \Delta y_i]$). The resulting deformed polygon mask is upsampled to input image dimensions via bilinear interpolation to finally obtain the output region mask.

2.3.3 Results

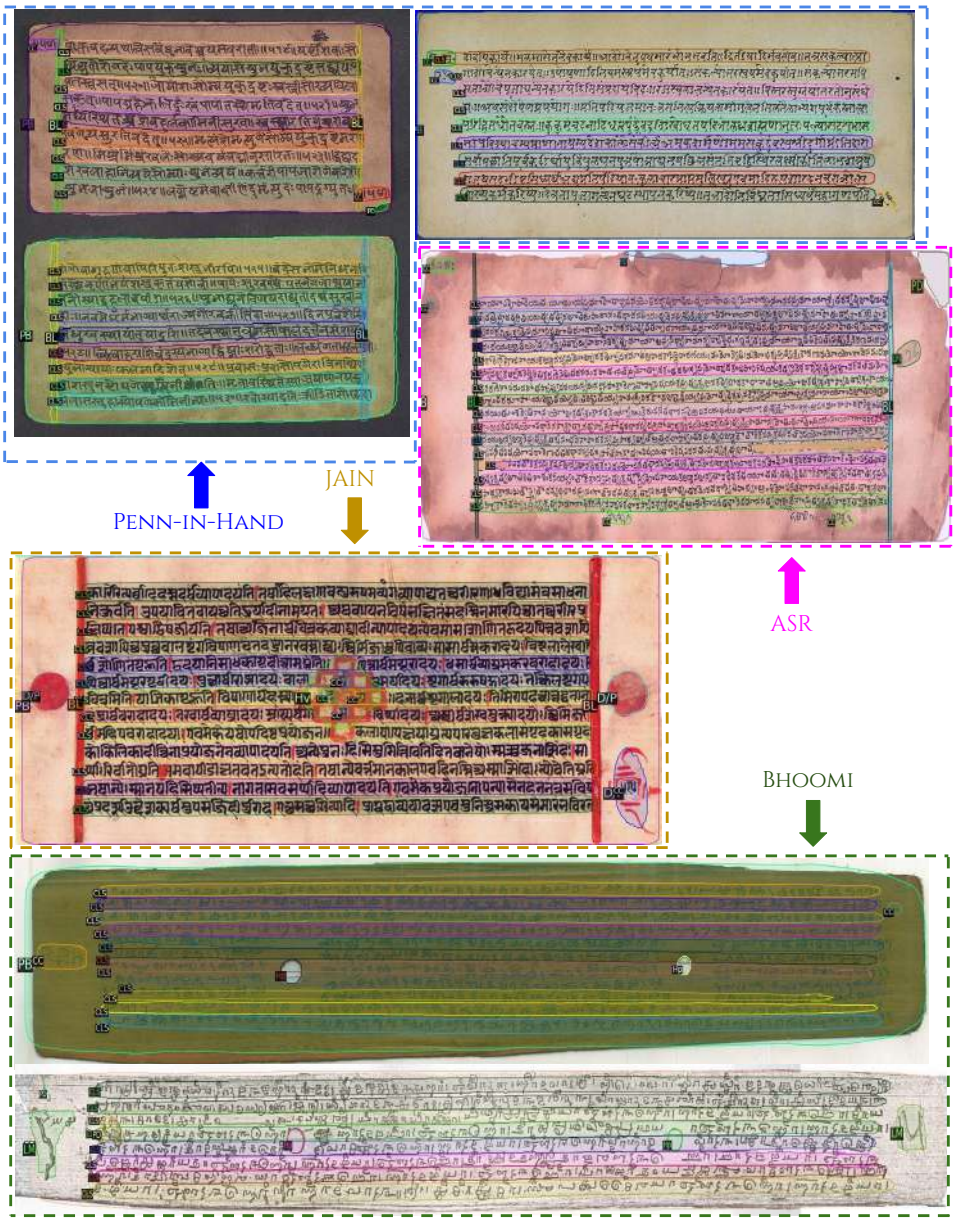


Figure 2.6: Set of Indiscapes2 dataset test images

2.3.4 Drawbacks

The above qualitative results show that predicted boundaries are still imprecise for the dominant region - character line segment and often miss out on characters at both ends of lines. Due to the global processing nature of the algorithm, some of the overlapping and dense lines are frequently not detected. In Fig(2.7), we highlight by observing more instances of omitted diacritics and characters.

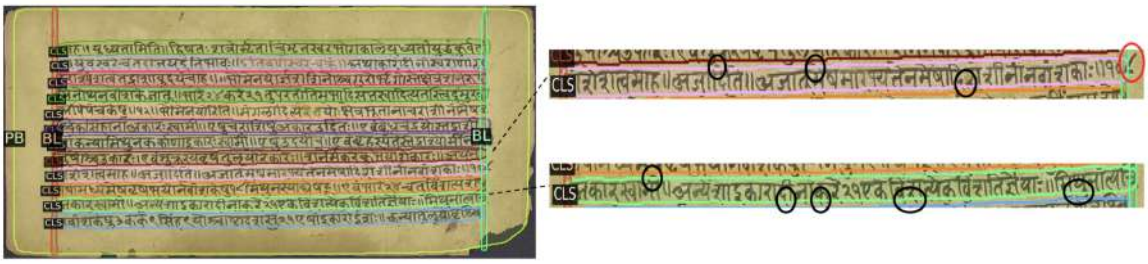


Figure 2.7: A Penn-In-Hand document featuring PALMIRA's text-line prediction lacks crucial diacritics and exhibits a bias toward predicting smooth, eclipse-like polygons.

2.4 BoundaryNet

After trying variants of Mask RCNN [1], [2] and understanding its constraints. A.Trivedi et al [44] proposed a novel , resizing free , semi-automatic approach for layout annotation in handwritten historical documents. To avoid any resizing of the input image , we ask the user the choose/draw a region of interest which is processed by an attention-guided skip network.The network optimization is guided via Fast Marching distance maps to obtain a good quality initial boundary estimate and an associated feature representation. These outputs are processed by a Residual Graph Convolution Network optimized using Hausdorff loss to obtain the final region boundary.

Mask CNN: The input to this block is an HxWxC crop of the image, which is selected by the user within the annotation tool. A UNet style with series of ResNet blocks is leveraged to avoid no spatial downsampling or upsampling.This is done to preserve crucial boundary information.Post the skip-attention backbone , we have two attached blocks - *Region Classifier* & *Mask Decoder*. In addition, features from the last residual block (Res-128) are fed to 'Region Classifier' sub-network which predicts the associated region class.Since input regions have varying spatial dimensions, we use adaptive average pooling to ensure a fixed-dimensional fully connected layer output. The input image is processed by an initial convolutional block with stride 2 filters before the resulting features are relayed to the backbone residual blocks. The spatial dimensions are restored via a transpose convolution upsampling within 'Mask Decoder' sub-network. These choices help keep the feature representations compact while minimizing the effect of downsampling.

Contourisation: The pixel predictions in the output from 'Mask Decoder' branch are thresholded to obtain an initial estimate of the region mask. The result is morphologically processed, followed by the extraction of mask contour. A b-spline representation of mask contour is further computed to obtain a smoother representation. M locations are uniformly sampled along the b-spline contour curve to obtain the initial set of region boundary points. For the subsequent GCN stage, these sampled points act are refined by the next stage.

Anchor GCN: From the previous stage, we get a initial estimate of contour of a region. The positional and appearance-based features of boundary points are progressively refine the region's boundary esti-

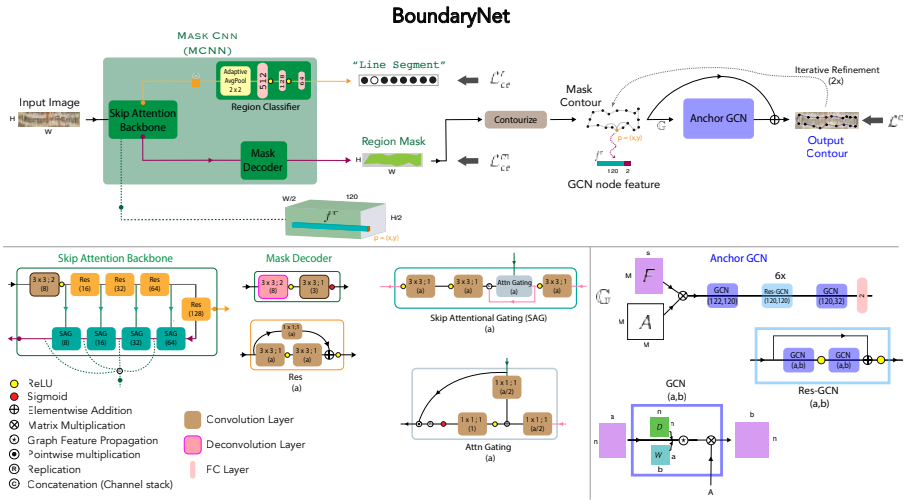


Figure 2.8: Detailed diagram of BoundaryNet Architecture

mate. Each node in the contour graph is a s-dimensional feature representation is comprised of (i) the contour point 2-D coordinates p = (x, y) (ii) corresponding skip attention features from MCNN. The input contour graph features are processed by a series of Res-GCN blocks sandwiched between two GCN blocks. The Anchor GCN module culminates in a 2-dimensional fully connected layer whose output constitutes per-point displacements of the input boundary locations. To obtain the final boundary, we perform iterative refinement of predicted contour until the net displacements are negligibly small by re-using GCN's prediction for the starting estimate at each iteration.

2.4.1 Results

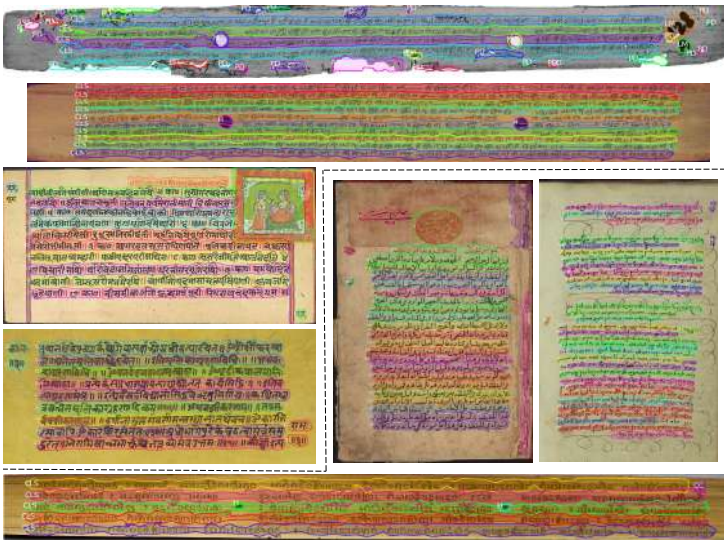


Figure 2.9: Qualitative Results of BoundaryNet on Indiscapes2 & other South-East Asian datasets.

2.4.2 Drawbacks

BoundaryNet surpasses its predecessors in performance, yet it tends to oversegment in character line segments, including regions from adjacent lines. While its precision in defining polygon boundaries has improved, resulting in fewer missing regions, BoundaryNet's reliance on user-input bounding boxes limits its effectiveness, especially in handling curved line segments. Additionally, its performance suffers in cases of orientation, as it erroneously includes other text regions in the final predicted polygon.

2.5 Conclusion

In studying different network architectures, a common problem arises: the segmentation masks tend to be imprecise for dominant classes (text lines). This happens because of the different methods used to resize images before feeding them into large-scale deep networks. Another important focus is on accurately connecting diacritics and enhancing the precision of character line instances. The future approaches must be able to generalize across multiple palm leaf manuscript datasets containing documents with varying scripts and text line densities.

Chapter 3

SeamFormer: High Precision Text Line Segmentation for Handwritten Documents

3.1 Motivation

Motivated by the need to address the challenges in document analysis that previous models like Palmira [1] and BoundaryNet [5].We introduce SeamFormer [4], a novel two stage text-line segmentation model (refer Fig. 3.2).SeamFormer offers full automation by seamlessly converting images to polygons, preserving the original image dimensions without the need for resizing (refer Fig. 3.1), thereby avoiding issues related to resizing described in Fig. 1.3 Ch. 1. Additionally, it places a strong emphasis on diacritics to effectively manage the complex vocabulary of South Asian and East Asian languages. These design choices collectively result in a highly precise text-line segmentation, ensuring more accurate document analysis.

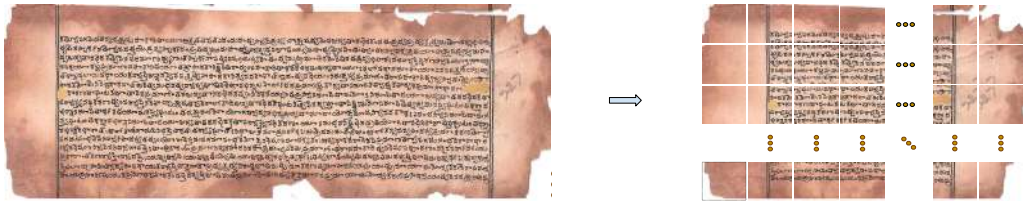


Figure 3.1: Full scale document image is divided into patch sizes of P x P, with overlapping patching mechanism that capturing neighbouring context.

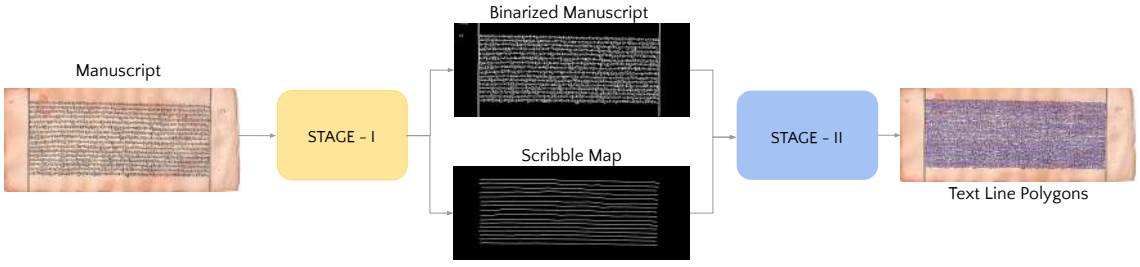


Figure 3.2: An outline of our SeamFormer pipeline highlighting Stage-I & Stage-II.

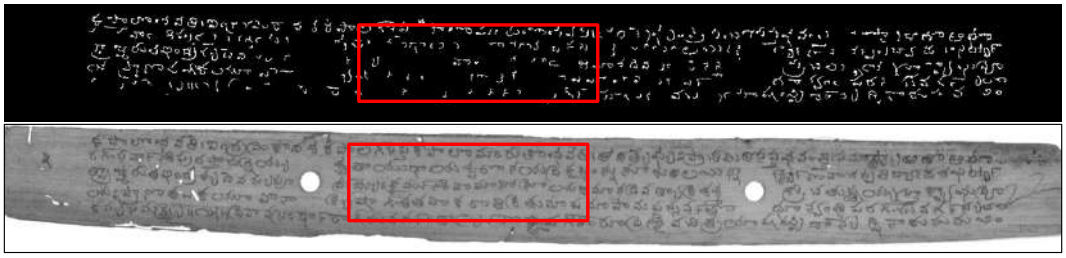
3.2 Overview

Given the input palm leaf manuscript image, our objective is to generate tight-fitting polygons enclosing each of the text lines. Our processing pipeline has two stages as show in Fig. 3.2 – *scribble generation* (Stage-I) and *text line polygon generation* (Stage-II) .As the input to the first stage , a patching algorithm is applied to the input image to divide into fixing size image patches 3.1. In the first stage, the manuscript image is processed by a deep network which generates coarse binary medial blobs for each individual text line and a binarized version of the image. The medial blobs are further processed to extract coarse spatial identifiers for each line termed as 'scribbles'. In the second stage, scribbles from first stage and custom-designed feature maps derived from binarized image are fed to a seam generation algorithm which generates the desired tight-fitting polygons enclosing the individual text lines.

3.3 Ground Truth Data Preparation

In the given pipeline, it should be noted that only Stage-I is learnable in nature and the associated architecture generates both binarisation and scribbles. Now , given that only have text-line polygons annotated for the Indiscapes2 dataset [1], algorithms were designed to generate binarisation and scribble ground truth.

(a) South East Asian - Sundanese Collection



(b) Indiscapes2 Bhoomi Collection

Figure 3.3: Examples of poor binarization ground truth in low contrast images across different palm leaf datasets.

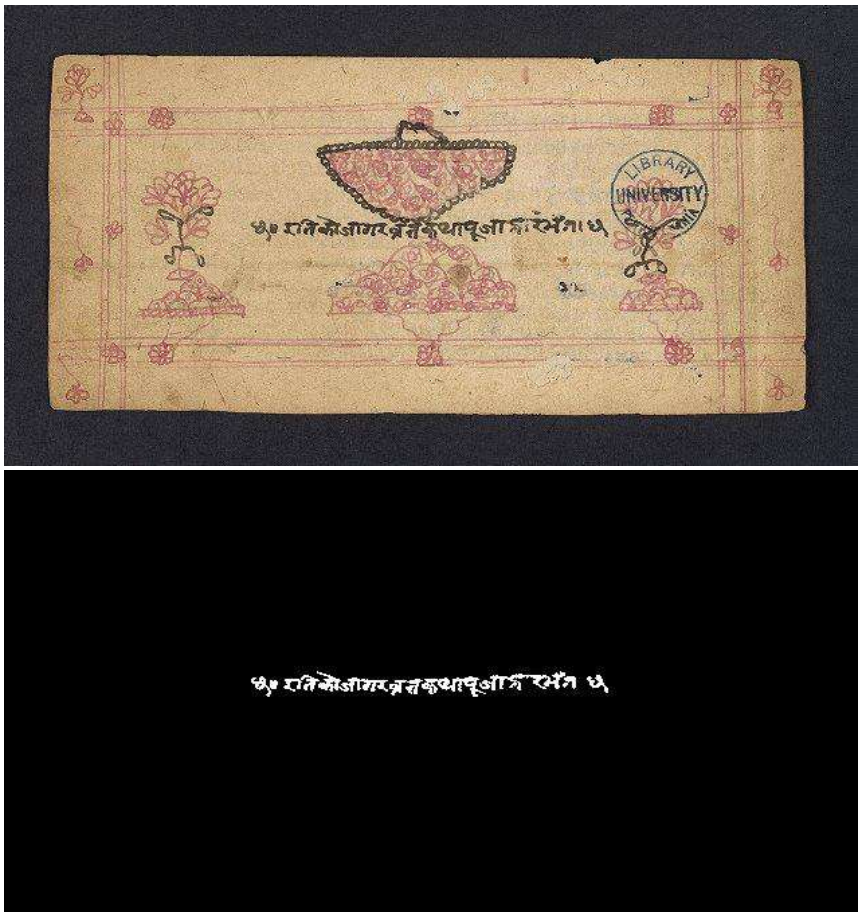


Figure 3.4: Example of Penn-In-Hand Illustration [1] image with all other elements like library markers , boundary lines and decorated being masked out to generate ground truth binarisation image.

3.3.1 Binary Map Ground Truth Generation

While our primary focus is on text segmentation, we address the text binarization issue as an auxiliary task in Stage-II of SeamFormer [4].To tackle this task, we retrain a state-of-the-art binarization network DocEntr[45], which has been previously trained large scale binarisation datasets such as [46], [47] etc .To adapt the model to our framework, we employ a combination classical image processing algorithm - Savoula-Niblack (SB) and the aforementioned state-of-the-art model to generate pseudo-binary ground truth annotations. In certain sub-collections (Fig. 3.3) the challenging nature of the documents necessitated manual annotations for accurate ground truth creation. Additionally, for documents containing multiple regions, such as boundary lines and decorators, we used polygonal annotations to mask out non-textual areas , as see in Fig. 3.4. Clean rich binarisation result is highly correlated with superior Stage-II performance. Despite lacking full-scale annotations for the binarization branch, we relied on the neural network to adapt the pre-trained binarization task for palm leaf manuscripts.

3.3.2 Scribble Ground Truth Generation

A scribble is a strikethrough across (left to right) that provides essential information about the local curvature of a text-line segment. Across most datasets, we encounter diverse sets of polygonal annotations such as tight-fit polygons, bounding boxes,binary masks of individual text-line characters and more. Thus scribble generation must be considered as a function of the existing annotation style of the text-line segment. While simple skeletonization may approximate a valid scribble for precisely annotated polygons, this may not be applicable to all annotation types. For loose polygonal ground truth annotations (such as bounding box), we employ connected component labeling (CCL) algorithm in conjunction with a binarized version of the image. Additionally, we dilate the text within the box to create a pseudo-blob-like structure. For each of these blobs, we generate skeletons and merge them based on proximity to form a connected spline.This entity effectively captures the turns and curves of the text within the text area.

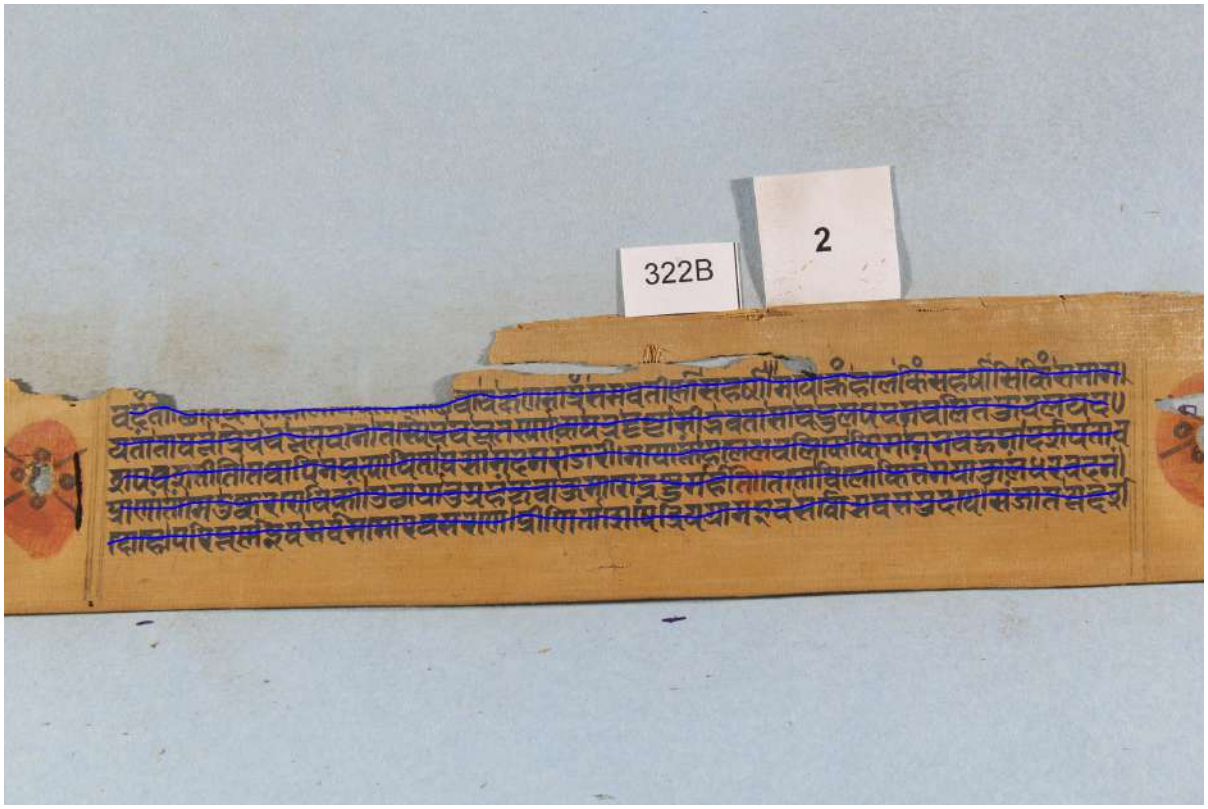


Figure 3.5: An image from Upamiti Bori (UB) collection to showcase how a scribble captures orientation of the text-line.

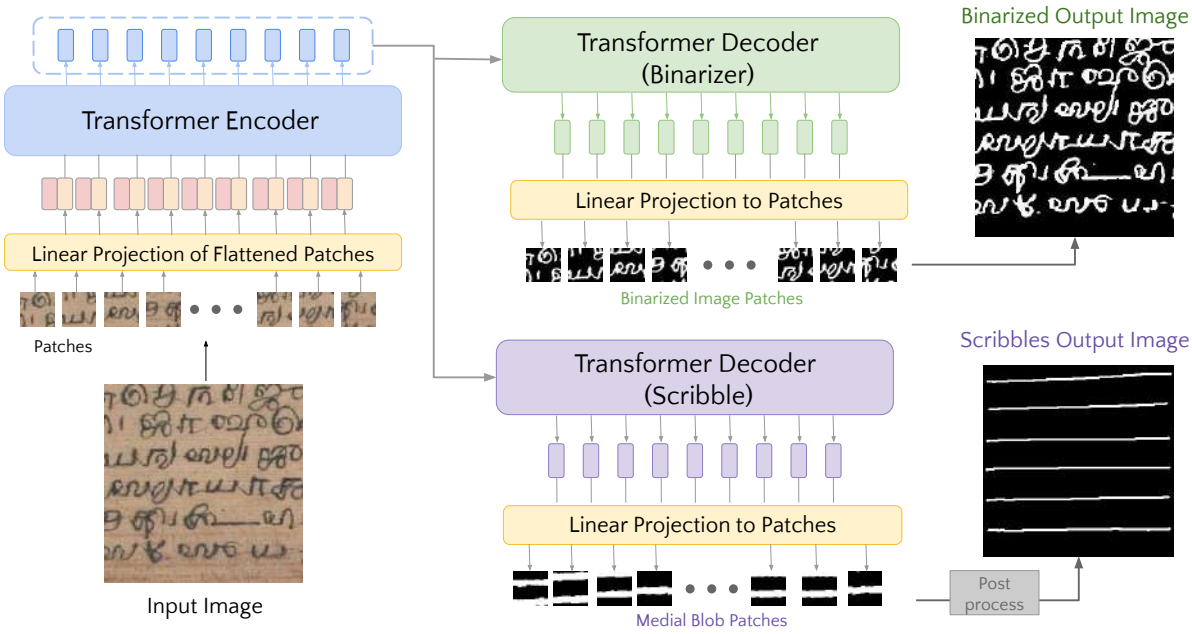


Figure 3.6: Stage-I : Multi-Task Vision Transformer Architecture

3.4 Stage I : Scribble & Binary Map Generation

For the purpose of generating the scribble and binary maps , we set up a multi-task variant of Vision Transformer (ViT) deep network architecture [48] to obtain two outputs - the binarized version of the input manuscript image and the medial blob masks for each text line 3.6. In a conventional Vision Transformer architecture, position-encoded patches of input image are processed within a Transformer [49] framework employing multi-head attention to obtain output patches. We extend the conventional setup to have two decoder branches. These branches output two sets of patches which are separately reassembled to obtain the binarized version of the input image and the medial blob masks binary image. The blob mask outputs are post-processed to extract thin medial axis-like structures which cut across the line – which we call as ‘scribbles‘.

3.4.1 Network Architecture

The multi-task architecture is designed based on the original Vision Transformer (ViT) proposed by Vaswani et al. (2017) [49]. The design choice of leveraging ViTs is to ensure there is resizing of inputs

It consists of an encoder-decoder architecture aimed at two variants of binary classification at the pixel level of an image, also known as binary segmentation. The patches embedding size and the number of transformer blocks are set depending on the model size. In our specific case, as discussed in Chapter 1, we are dealing with historical datasets characterized by extremely high aspect ratios, with dimensions often in the order of thousands. To address this challenge, we partition the document image into smaller sub-images of size 256 x 256 pixels, which serve as input to the transformer’s encoder. To

ensure optimal sampling, we incorporate a 25% overlap between adjacent sub-images.

Encoder In the encoding stage (left part of Fig. 3.6), given an 256x256 input sub-image, we divide it into a set of 16x16 patches. Then, we embed these patches to obtain the tokens and add their positional information. After that, a number of transformer blocks is employed to map these tokens into the encoded latent representation. These blocks follow the same structure as [49], composed of alternating layers of multi-headed self-attention and multi-layered perceptron (MLP). Each of these blocks are preceded by a LayerNorm (LN) , and followed by a residual connection.

Decoder The decoder part consists of a series of transformer blocks (having the same number as the encoder blocks) that take as an input the sequence of outputted tokens from the encoder. These tokens are propagated in the transformer decoder blocks, and then projected with a linear layer to the desired pixel values. This makes each element of the output correspond to a vector representing a flattened patch in the output image. The ground truth pixel values are obtained by dividing the ground truth (GT) clean image into patches (in the same way as the input document image) and flattening them into vectors.

Because both tasks involve binary classification, where regions of text are assigned a value of 1 (indicating high importance) while the remaining background elements, such as decorations,library markers,etc are ideally suppressed or assigned a value of 0, it is imperative to employ a shared encoder. This shared encoder is tasked with learning to focus on the text contained within the sub-image, as it operates on the same input image patch. Both of the decoders – scribble branch and binarizer have different learning styles and dedicated loss functions for optimal performance.

3.4.2 Post-Processing Block

As seen in Fig 3.6, we focus on how the text blobs produced in Stage-I. The design of post-processing block was on the premise that scribble maps were thick blobs that needed to be refined into thin line like structures. A challenge during this process is that these blob like structures often are too thick and completely vary with the width of the text. This was a repeating pattern despite training with thin scribble structures with thickness = 1. To combat this , the post processing block has multitude of document specific parameters to decrease this effect. As described in Algorithm 1, we start out by thresholding both the scribble and binary logits, with a fixed threshold of 0.5. To clean the scribble map from additional blob structures which might be predicted due to elements like library markers or decorators , etc we leverage the binary map. Within the binary map , we dilate the text with small 3x3 dilation kernel, and then perform a bitwise AND operation with the scribble map. This ensures that there are disconnected blob structures only in the text region. We then connect all these mini-blob structures to get the largest contour , and then extract a minimum fit bounding box. We apply our further processing only within this region. We apply Distance Transform on the thresholded scribble map to generate skeleton structures of these thick medial blobs. We perform bitwise AND operation between this map and dilated the binary output. As a result , we obtain map with small text blobs we join them horizontally form long medial blob and then generate a convex hull around to smooth the finer curvature of the polygon. Subsequently, we apply skeleton pruning techniques to remove spurious branches and

extract a clean medial fragment for each blob within the image. In case of fragments of scribbles, we group them based on distance thresholding technique as a function of its horizontal level. We then have a clean of scribbles for every text-line present in the document and we proceed to Stage-II.

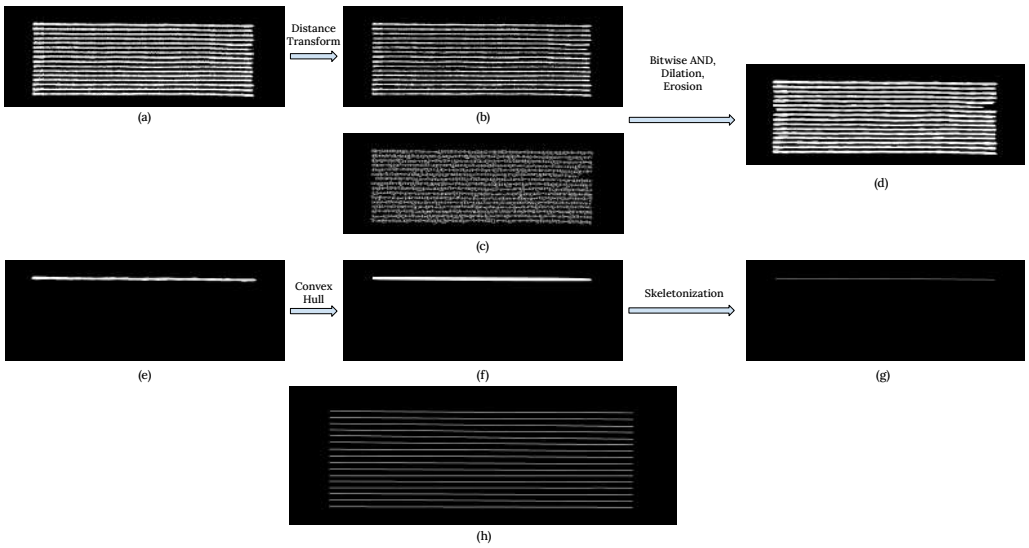


Figure 3.7: Stage I: Post Processing Module - see Sec. 3.4 and Algorithm 2 for details. Figure (a), (c) is the medial blob output, binarisation output respectively from the transformer. Figure (b) is the distance transformed result of figure (a). Bitwise and operation result is obtained using figure (b) and (c). Then dilation and erosion is applied on it to get figure (d). Figure (e) represents a single medial blob for simplicity. Figure (f) is obtained by obtained convex hull of figure (e). Figure (g) is obtained by performing skeletonization on figure (f). Figure (h) represents the final thin scribbles obtained after post-processing all the polygons.

3.5 Stage II: Text Line Polygon Generation

From the previous stage , we have our dedicated set of scribbles and binary map. We have designed a scribble-conditioned seam generation algorithm with custom maps , inspired from the [26]. This stage involves two sub-stages – Feature Map Generation and Scribble-conditioned Seam Generation (see Fig. 3.8). For each scribble, we first generate a corresponding pair of *pseudo-scribbles* which are used at later stages of the pipeline (Sec. 3.5.1). Next, the scribbles are overlaid on binarized input image and the resulting scribble-overlaid image is used to create custom feature maps (Sec. 3.5.1.1). These feature maps are used as input to a seam generation procedure to generate the desired high-precision polygons enclosing the text lines (Sec. 3.5.1.2).

3.5.1 Pseudo-scribble generation

As the first step, we sort the scribbles by the y-coordinate of the left-most point to obtain the sequence of scribbles S in a top-to-bottom order. Let $s_i \in \mathcal{S}$ be a scribble. Let μ_{s_i} be the average of all y-

Algorithm 1 Post-Processing Algorithm (Sec. 3.4.2)

▷ **Input** Binary Map B & Scribble Map $\mathcal{S}$ from Stage I & Threshold Parameter(θ)
▷ **Output** Set of Scribbles $\mathcal{P}$
$B \leftarrow$ THRESHOLD(B, θ)
$S \leftarrow$ THRESHOLD(S, θ)
▷ Generate Text Box
$TextBoxMask \leftarrow$ TEXTBOXDETECTION(B)
▷ Apply Distance Transform
$S' \leftarrow$ DISTANCETRANSFORM(S)
▷ Removing Unnecessary Blobs
$U \leftarrow$ BITWISEAND$(S', B, TextBoxMask)$
▷ Text Dilation & Blob Detection
$F \leftarrow$ TEXTBLOBGENERATION(U)
▷ Combining small blobs to larger parent blobs
for b **do** in $\mathcal{F}$: ▷ For each blob
    $b' \leftarrow$COMBINEBLOBSONHORIZONTALLEVEL$(b, distanceThreshold)$
    ▷ Generate Scribble for the resulting blob
    $b'' \leftarrow$ CONVEXHULLCREATION(b')
    $s \leftarrow$ GENERATESCRIBBLE(b'')
    $\mathcal{P} \leftarrow \{s\} \cup \{P\}$
end for
return $\mathcal{P}$

coordinates of the scribble s_i's pixels. Let $\mu_{s_{i+1}}$ be a similar average for the neighboring scribble. The vertical offset between the scribble pair can be defined as $d(s_i, s_{i+1}) = \left|\mu_{s_i} - \mu_{s_{i+1}}\right|$. Let $\overline{d(\mathcal{S})}$ denote the average across all such vertical offsets within the set of scribbles. Define $\theta = \overline{d(\mathcal{S})} + \delta$ where δ is a fixed offset. For each scribble s, the upper pseudo-scribble (u) and lower pseudo-scribble (l), are obtained by vertically translating s by $+\theta$ and $-\theta$ pixels respectively – see the block 'Pseudo scribbles' which is part of 'Scribble-Conditioned Seam-Generation' (shaded blue) in Fig. 3.8.

3.5.1.1 Feature Map Generation

Gradient Map (GM): This feature map is obtained as the gradient magnitude map of the scribble-overlaid image. Using this map creates a high energy barrier between edges of characters in the text line and the background area immediately surrounding them. Employing this map in the subsequent seam generation stage enables seams to align closely with text letter boundaries, resulting in tight-fitting polygons around the text lines (ref. $\mathcal{GM}$ in Fig. 3.8).

Smoothing Map (SM): This feature map is obtained by applying a blur kernel on the scribble-overlaid image. Using this map increases the energy at horizontal inter-character text gaps and ensures that seams do not cut through the text (ref. $\mathcal{SM}$ in Fig. 3.8).

Diacritic Map (DM): This novel feature map specifically tackles the problem of diacritics not being enclosed within the polygons of corresponding parent text lines - see Fig. 3.9. We first isolate the

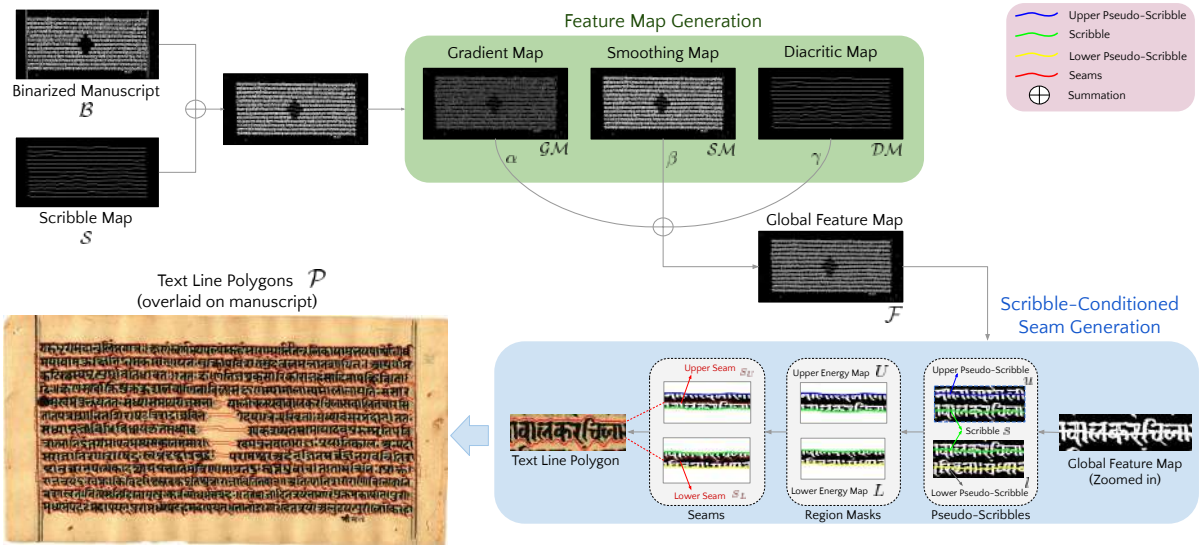


Figure 3.8: Stage II: Text Line Polygon Generation Module - see Sec. 3.5 and Algorithm 2 **??** for details.

region around each text line with the help of upper and lower pseudo-scribbles as the demarcations. We overlay the corresponding scribble on the parent text-line and perform connected components analysis. This operation divides the components into three major groups: components connected to parent-line, disconnected diacritics and background noisy elements. We discard noise based on an area threshold. For each diacritic component, we connect its centroid and parent scribble via a perpendicular line. In effect, this line creates an energy barrier which forces the boundary generated during seam generation to move around the diacritic instead of separating the diacritic and its parent text line (ref. $\mathcal{DM}$ in Fig. 3.8). The utility of Diacritic Map is illustrated in Fig. 3.9. The neighborhood of a text line often contains text fragments from adjacent lines due to the uneven handwritten line orientation and dense handwriting. Our construction of the Diacritic Map actively prevents the neighbouring text fragments from being picked up along with the diacritics.

The weighted combination of the above feature maps forms the final global feature map, i.e. $\mathcal{F} = \alpha\ \mathcal{GM} + \beta\ \mathcal{SM} + \gamma\ \mathcal{DM}$. Figure 3.10 illustrates the importance of using scribbles and the proposed combination of energy maps. It is important to note that unlike some of the existing seam-based approaches [27], we generate the feature map only once for the input image.

3.5.1.2 Scribble-conditioned Seam Generation

For each scribble s, the paired end-points of the scribble and its corresponding upper pseudo-scribble u are connected to obtain an enclosed upper region U - see the block 'Region Masks' which is part of 'Scribble-Conditioned Seam-Generation' (shaded blue) in Fig. 3.8. The region's mask is applied to global feature map $\mathcal{F}$ and cropped to obtain the upper region feature map $\mathcal{F}_U$ for the scribble. To constrain the seams to lie within the masked portion, feature map values outside the mask are set to a fixed 'high energy' value. The upper region feature map is used during seam generation [26].

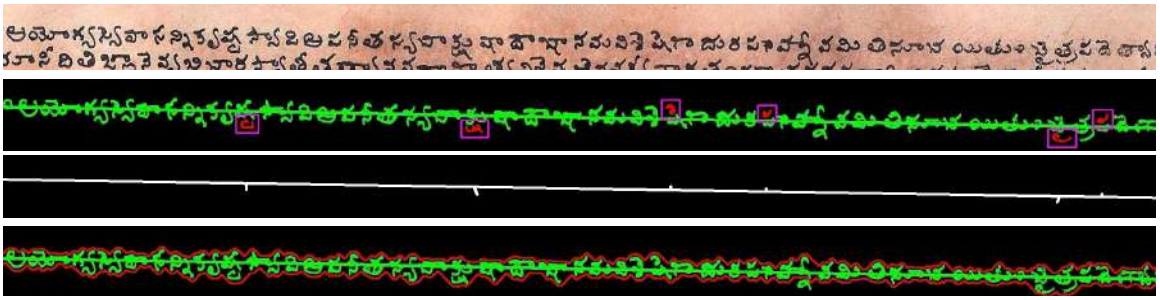


Figure 3.9: Diacritic Map (Sec. 3.5.1.1) - (a) A text line from a palm leaf manuscript, (b) the reference text line is shown with the scribble overlaid. Pixels in green denote the text line connected by the scribble and pixels in red inside pink bounding boxes denote the corresponding diacritics of the parent text line (c) Diacritic Feature Map - note the tiny strokes extending out of the scribble to connect the diacritics with the main text line (d) final red seams enclosing the text line as a result of using the Diacritic map during seam generation - note that the aforementioned diacritics have been brought inside the enclosing seams.

Figure 3.10: (a) A fragment from the top portion of a manuscript (b) Seams generated with Gradient and Smoothing Map, but without using scribble – the upper line boundary is missing (c) Seams when scribble is also added – upper line boundary is obtained, but diacritics are missed (d) Seams when Diacritic Map is also included – line boundaries properly enclose text and associated diacritic components.

For a $M \times N$ image, a horizontal seam R is a connected sequence of pixels and can be defined as $R = (x_i, y_i); i = 1, 2, \ldots r, 1 \leqslant x_i \leqslant N, 1 \leqslant y_i \leqslant M$ where $x_1 = 1, x_r = N$ and $|x_i - x_{i-1}| \leqslant 1, i = 2, 3, \ldots r$. The 'energy cost' of the seam is defined as $U(R) = \sum_{i=1}^{r} \mathcal{F}_U(x_i, y_i)$. The seam with the minimum cost is defined as $S_U = \arg\min_R U(R)$ and is found using dynamic programming. In this context, feature map $\mathcal{F}_U$ has been constructed such that the minimum energy seam corresponds to tight upper boundary of the associated text line. Additionally, to enhance the tight-fit characteristic of the seam, we induce a bias in choosing the lowest energy path. During the seam propagation step, we greedily pick the lowest x or y coordinate value among potential energy paths. This choice results in energy seams circumscribing the character components tightly. A similar procedure as above is repeated with the lower pseudo-scribble l to obtain a tight lower boundary seam S_L for the text line. These seams (S_U, S_L) are connected at their paired endpoints to obtain the final high precision polygon P enclosing the text line.

Algorithm 2 Scribble-Conditioned Text Line Polygon Generation (Sec. 3.5.1.2)

▷ **Input** binaryImage B and set of scribbles $\mathcal{S}$ from Stage I (Sec. **??**)
▷ **Output** Set of text line polygons $\mathcal{P}$
$\theta \leftarrow$ COMPUTEGAP($\mathcal{S}$) ▷ Obtain interline gap using inter-scribble gap statistics
▷ Feature Map Generation
$\mathcal{GM} \leftarrow$ GENERATEGRADIENTMAP($B, \mathcal{S}$)
$\mathcal{SM} \leftarrow$ GENERATESMOOTHINGMAP($B, \mathcal{S}$)
$\mathcal{DM} \leftarrow$ GENERATEDIACRITICMAP($B, \mathcal{S}$)
$\mathcal{F} \leftarrow$ GENERATEGLOBALFEATUREMAP($\mathcal{GM}, \mathcal{SM}, \mathcal{DM}$)
▷ Scribble-conditioned Seam Generation
for s in $\mathcal{S}$ **do** ▷ For each scribble
    $u, l \leftarrow$GENERATEPSEUDOSCRIBBLES(s, θ)
    ▷ Generate upper seam
    $U \leftarrow$ GETREGION(s, u)
    $\mathcal{F}_U \leftarrow$ GETCROPPEDFEATUREMAP($U, \mathcal{F}$)
    $S_U \leftarrow$ GENERATESEAMS($\mathcal{F}_U$)
    ▷ Generate lower seam
    $L \leftarrow$ GETREGION(s, l)
    $\mathcal{F}_L \leftarrow$ GETCROPPEDFEATUREMAP($L, \mathcal{F}$)
    $S_L \leftarrow$ GENERATESEAMS($\mathcal{F}_L$)
    ▷ Generate the final text line polygon
    $P \leftarrow$ GENERATELINEPOLYGON(S_U, S_L)
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{P\}$
end for
return $\mathcal{P}$

It is important to note that the scribble generated in Stage-I determines the sub-image region in which seam generation operates. Confining seam generation by using scribble-based masks helps produce compact enclosing boundaries (see Figure 3.10).This is unlike other seam-based methods which generate seams that go beyond actual extent of the text line. Algorithm 2 outlines the procedure for scribble-conditioned text line polygon generation.

3.6 Training & Evaluation

Stage-I: For the ViT network, we use 256×256 overlapping manuscript patches with appropriate padding. Resampling is used to overcome the imbalance between text and empty (non-text) patches. For training the binarizer branch for South-East Asian datasets, we use the binary dataset from Challenge A of the ICFHR 2018 contest [3]. For other datasets, we use Sauvola-Niblack binarisation [50, 51] as the ground truth. We initialize the binarization branch with pre-trained weights [45]. The learning rate is initialized to 0.05 and is decayed by Pytorch's learning scheduler, ExponentialLR with $\gamma = 0.8$. For training both of these branches we leverage the L2 loss. We adopt a training procedure where every individual branch is trained separately, while the other branch's weights are frozen. The optimizer

used is stochastic gradient descent with $\gamma = 0.1$ and momentum of 0.9. We perform data-parallel optimization distributed across 2 GeForce RTX 2080 Ti GPUs for 40 epochs, with a fixed batch size of 4. We use random rotation augmentation $\alpha \in (-30, 30)$ to improve performance for non-axis oriented manuscripts. To tackle varied manuscript background textures and noise, we apply Gaussian Noise, AdvancedBlur, RandomColor, RandomFog, RandomBrightness and HueSaturations augmentations [52]. For post-processing, we apply erosion filters - a horizontal rectangular kernel 1×11 thrice, followed by a 1×1 dilation to separate any overlapping medial blobs. These blobs undergo a skeletonization procedure followed by pruning to remove any spurious branches with a minimum area threshold of 100 pixels. The post-processing is robust and does not need to be changed across datasets or approaches.
Stage-II: The offset for pseudo-scribble generation δ is set to 5. In the feature map generation pipeline, we use the standard 3×3 Sobel kernel for Gradient Map. We apply a Gaussian blur kernel of 15×11 for high spatial coverage within the image to compute the Smoothing Map. The weights for various feature maps are empirically set to $\alpha = 0.4$ ($\mathcal{GM}$), $\beta = 0.6$ ($\mathcal{SM}$) and $\gamma = 1.0$ ($\mathcal{DM}$). The global feature map is normalised to $[0, 1]$ before the seam generation process.

3.7 Qualitative Images

In this section, we present examples of outputs generated by SeamFormer when applied to documents from South East Asia and a selection of Indic Manuscripts. We highlight the importance of precise polygon generation in SeamFormer, contrasting it with other methods that frequently produce smoother, curved polygons. Additionally, SeamFormer stands out as the first model to prioritize the inclusion of diacritics, employing a specialized heuristic approach to handle them effectively.

3.8 Quantitative Results

Refer to Table 3.1 on the following page to understand how SeamFormer stands with existing approaches on benchmark datasets.

3.9 Conclusion

We introduce SeamFormer, a novel approach for high precision text line segmentation in handwritten documents. Instead of a monolithic framework, we tackle the challenge of text line segmentation using a divide-and-conquer two stage approach. The first stage generates medial line 'scribbles' which provide crucial information about the curvature of the text line and a binarized version of the input image. In the second stage, these scribbles and custom-designed feature maps derived from the binarized image are fed to a seam generation algorithm which generates the desired tight-fitting line polygons. Our approach is a resizing-free method. As a result, text line gaps are not distorted or aliased, leading to significantly better results. Our novel inclusion of Diacritic Map in the second stage ensures complete and correct inclusion of diacritics within the predicted polygon. Also, pseudo-scribbles are a key

Table 3.1: Comparison of SeamFormer with existing approaches on benchmark datasets (Sec. 3.8).

	Indiscapes2[41]	KGathaM	Bali[53]	Sunda[54]	Khmer[55]
			IoU ↑		
MMRCNN [41]	0.55	0.34	0.23	0.28	0.28
Palmira [1]	0.76	0.69	0.42	0.68	0.45
Doc-UFCN [35]	0.16	0.12	0.08	0.23	0.10
dhSegment [37]	0.34	0.12	0.03	0.12	0.08
LCG [30]	0.37	0.20	0.12	0.12	0.18
DocExtractor [38]	0.10	0.17	0.01	0.02	0.04
SeamFormer	**0.78**	**0.84**	**0.66**	**0.77**	**0.69**
			HD ↓		
MMRCNN [41]	447.58	855.76	2106.30	1147.30	1760.48
Palmira [1]	73.32	57.84	1699.58	130.34	1190.95
Doc-UFCN [35]	339.30	238.87	1873.00	630.79	2552.26
dhSegment [37]	295.58	216.79	2232.90	394.16	1560.45
LCG [30]	207.76	346.93	797.51	367.60	496.31
DocExtractor [38]	806.17	1423.26	3552.19	1865.25	3987.37
SeamFormer	**21.91**	**16.05**	**48.86**	**32.18**	**48.37**
			AvgHD ↓		
MMRCNN [41]	57.13	132.50	302.59	145.07	270.47
Palmira [1]	7.29	2.74	224.79	6.50	203.59
Doc-UFCN [35]	70.04	49.16	319.06	98.55	481.19
dhSegment [37]	60.33	43.60	415.24	66.77	319.57
LCG [30]	16.82	29.72	95.18	39.65	44.50
DocExtractor [38]	149.29	219.68	778.60	331.00	898.16
SeamFormer	**0.65**	**0.25**	**2.53**	**1.01**	**2.39**
			HD95 ↓		
MMRCNN [41]	355.74	702.45	1766.12	918.85	1449.68
Palmira [1]	42.47	21.49	1393.15	49.06	1019.12
Doc-UFCN [35]	304.73	214.35	1628.43	520.38	2271.27
dhSegment [37]	262.83	192.33	1967.72	329.84	1380.09
LCG [30]	99.77	197.94	390.21	231.38	191.88
DocExtractor [38]	595.61	1084.01	3255.44	1656.47	3654.05
SeamFormer	**4.59**	**1.96**	**19.49**	**7.77**	**18.83**

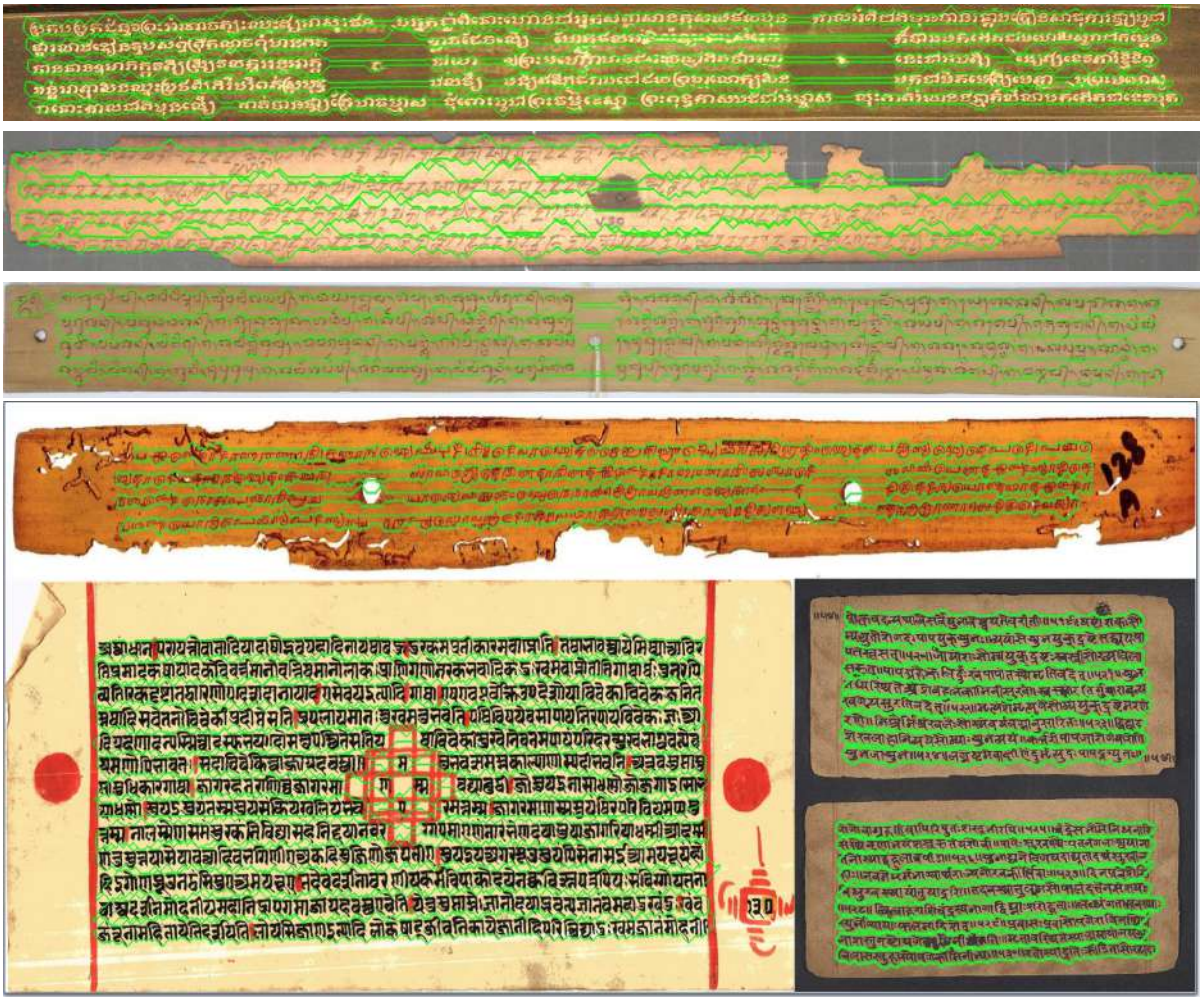


Figure 3.11: SeamFormer predictions on South-East Asian Manuscripts [3]: Khmer (top), Sundanese (middle), and Balinese (bottom). Predictions on Indiscapes2: Bhoomi, Penn-In-Hand, and Jain manuscripts (bottom - clockwise) [1].

innovation in our approach. The pseudo-scribbles serve as energy barriers during seam generation and ensure the seams do not cross the text line's spatial extents. The pseudo-scribbles also prevent the seams from deviating too much from the reference line unlike some existing approaches. Overall our design choices attribute to important aspects of text-line segmentation specific to Indic/South-East Asian documents.While progress has been made in text line segmentation, there remain areas for enhancement in both Stage-I and Stage-II to broaden its application to larger datasets. In Stage-I, solving scribble generation as a binary segmentation task increases the need for post-processing and can lead to significant overlap between adjacent text blobs, thus adversely affecting performance during seam generation. In Stage-II, the accuracy of the diacritic map depends heavily on the accuracy of scribbles and binary maps, with inaccuracies resulting in numerous false positives. Both stages are influenced by document collection-specific parameters, highlighting the necessity for empirical testing.

Chapter 4

LineTR : Unified Text Line Segmentation for Challenging Palm Leaf Manuscripts

4.1 Motivation

In this work , we try to approach and resolve challenges faced in SeamFormer [4] and align to a more universal approach for precise text line segmentation. As highlighted in previous chapter, each of the stages within the pipeline can be enhanced and become less parameteric in nature. In Fig 4.1, describes how relying on binarisation for scribble generation can be costly especially in highly dense documents. Throughout this chapter, we present comparative diagrams at different stages, illustrating the challenges encountered with SeamFormer and how LineTR effectively resolves them.

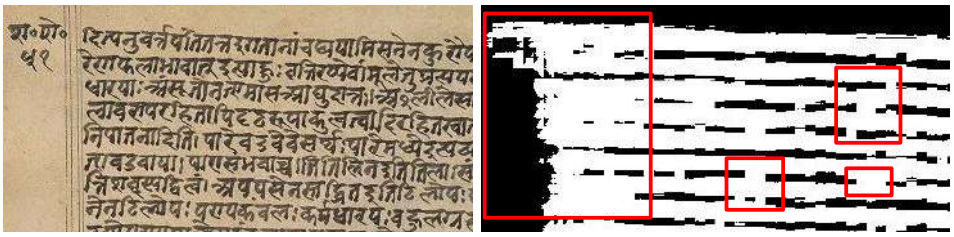


Figure 4.1: Issues with SeamFormer's [4] raw scribble map output on a dense Indic manuscript [1]. SeamFormer formulates scribble prediction as a per-pixel binary classification task, leading to extremely noisy predictions. Highlighted regions in the image cannot be post-processed to obtain distinct scribbles because of extreme merging.

4.2 Overview

Motivated by several needs, we propose LineTR, a unified, highly adaptive and precise text segmentation approach for challenging historical manuscripts. It is designed as a two-stage pipeline similar to SeamFormer [4].To begin with, overlapping patches are sampled from input image in a context-adaptive manner. The first stage (refer Fig. 4.2) consists of [i] a novel DETR-style [56] deep network which generates parametric representations of text strike-through lines (scribbles) (refer Fig. 4.2) and [ii] a novel hybrid CNN-transformer network which generates a text energy map (refer Fig. 4.5). A dataset-

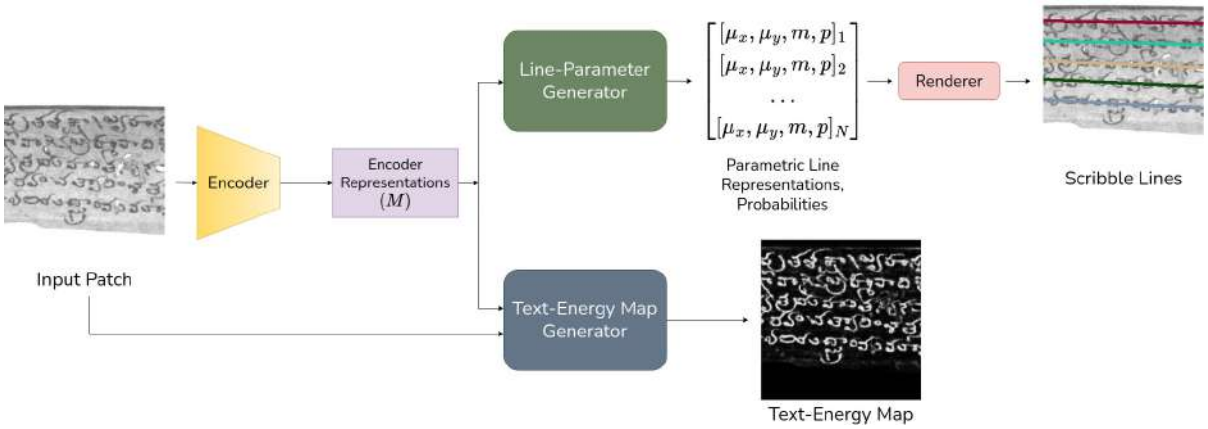


Figure 4.2: LineTR Stage-I Pipeline (Sec. 4.4)

agnostic and robust post-processing procedure is applied on patch-level predictions from Stage-1 to obtain document-level scribbles. The scribbles and text energy map are used within an enhanced seam generation framework (the second stage) to obtain highly precise polygons enclosing the manuscript text lines. As part of this work , three existing dataset pooled from private and public collections of Indic and South-East Asian heritage were annotated. The annotation was performed with a modified version of HINDOLA [5] tool with SeamFormer [4]'s Stage-II integrated to it (refer Appendix A).

4.3 Network Architecture

We adopt a two stage approach to predict text line polygons [4]. *Stage-1* (see Fig. 4.2): First, the input image is split into overlapping contextual patches of various sizes (Sec. 4.5.1). Each patch is processed by a deep network which predicts (a) parametric representations of text strike-through lines (scribbles) and (b) a continuous binary energy map (Sec. 4.4). The per-patch outputs are merged using an adaptive, data-agnostic post-processing module to obtain a global scribble map and a global binary energy map corresponding to the entire image (Sec. 4.5.2). *Stage-2:* The global maps from *Stage-1* are processed using a seam generation algorithm to obtain tight-fitting polygons enclosing the text lines (Sec. 4.6).

4.4 Stage I

In this stage, the input patch is first processed by a shared encoder. The encoder representations (M in Fig. 4.2) are fed to the Line-Parameter Generator which outputs parametric representations of scribble segments in the patch. Next, we describe the individual components of *Stage-1* pipeline. As mentioned before , we approximate scribbles are straight lines and approach it as a geometrical problem. We represent each scribble line using three parameters (Fig. 4.3) – μ_x, μ_y and m where (μ_x, μ_y) represents the mid-point of the scribble segment and m represents the slope. μ_x and μ_y are normalized w.r.t patch

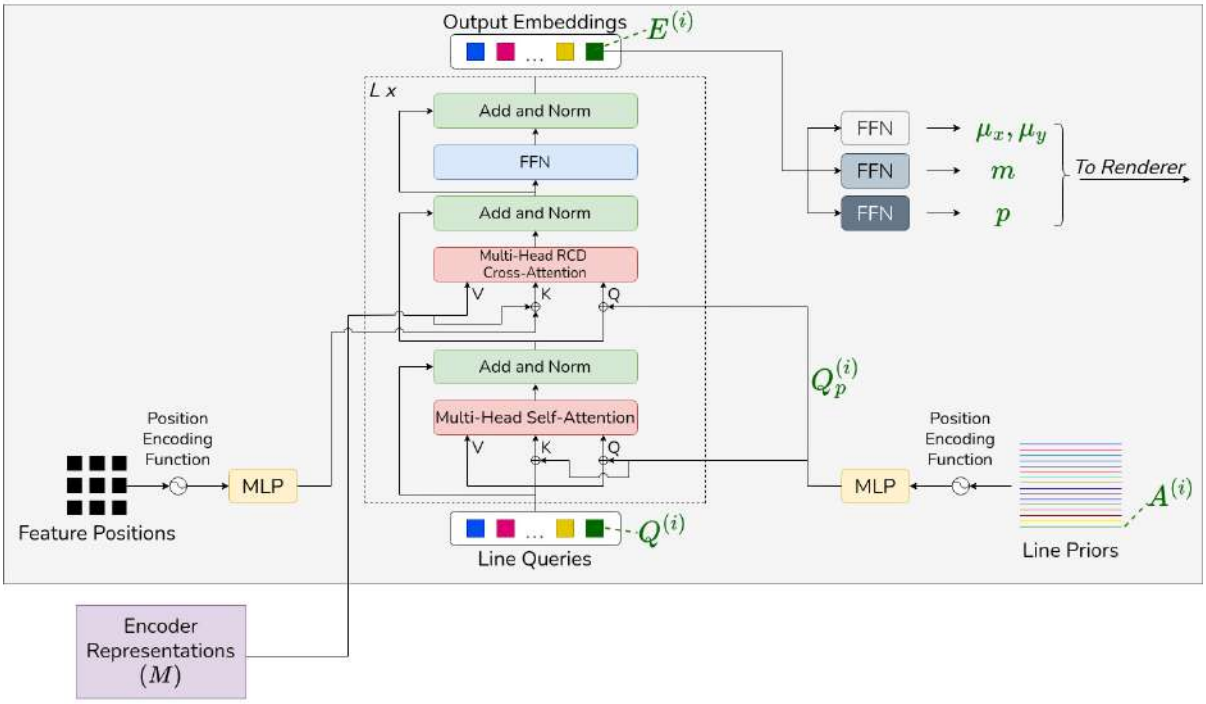


Figure 4.3: Line-Parameter Generator

dimensions by dividing with image width and height respectively so that $\mu_x, \mu_y \in [0, 1]$. Hence, we design out Stage-I to predict these parameters efficiently.

4.4.1 Backbone

Instead of the proposed CNN based backbone for image feature extraction proposed in DETR [56], we opt for Vision Transformer (ViT) [48] which outputs a feature map $M \in \mathbb{R}^{H \times W \times C}$, where C is the channel dimension and H, W the spatial dimensions.

4.4.2 Line-Parameter Generator

We introduce a novel DETR-style [56] framework to predict parametric representations for each scribble line. A set of N learnable embeddings, which we call 'Line Queries' are fed to a transformer decoder (see Fig. 4.3). Within the decoder, these line queries are processed along with encoder representations M to obtain latent representations for the scribble lines ('Output Embeddings' in Fig. 4.3). These latent representations are transformed via lightweight feed-forward networks (FFN) to obtain scribble line parameters μ_x, μ_y, m and associated probabilities p. Next, we describe some key components of the transformer decoder.

Line Queries: We first define 'Line Priors'. These are N horizontal lines distributed uniformly throughout the image (see Fig. 4.3). Formally, the i-th line prior $A^{(i)}$ is parameterized as $\mu_x = 0.5, \mu_y = i/N, m = 0$. We define positional query $Q_p^{(i)}$ as the positionally encoded and transformed version of

$A^{(i)}$. Each line query $Q^{(i)}$ is first initialized to zero. At the query and key input of every attention layer, we add positional query $Q_p^{(i)}$ (see Fig. 4.3).

Self Attention: There are two types of attention in the transformer decoder – self attention and cross attention [49]. The self attention is applied within the *Line Queries* Q (defined previously). Note that the positional queries are added to the line queries before computing the attention scores.

Cross Attention: The outputs of *Self Attention* are fed to the cross attention module. The attention weights in classical cross attention are given by $\Psi = \text{softmax}\Big(\frac{QM^T}{\sqrt{d_k}}\Big)$, where $Q \in N \times C$ are *Line Queries*, $M \in HW \times C$ stands for encoder representations. Now, attention weights $\Psi \in N \times H \times W$ are often many in number. The large size of Ψ often leads to slow convergence in DETR-style frameworks [56, 57, 58]. To counter this, we adopt a decoupled row-column-based attention proposed by [57]. Effectively, the number of attention weights in this attention scheme are reduced to $N(H+W)$ from NHW, which greatly helps in faster and better convergence.

In our work , the outputs of *Self Attention* are fed to the cross attention module. The attention weights in classical cross attention are given by $\Psi = \text{softmax}\Big(\frac{QM^T}{\sqrt{d_k}}\Big)$, where $Q \in N \times C$ are *Line Queries*, $M \in HW \times C$ stands for encoder representations. Now, attention weights $\Psi \in N \times H \times W$ are often many in number. The large size of Ψ often leads to slow convergence in DETR-style frameworks [56, 57, 58]. To counter this, we adopt a decoupled row-column-based attention mechanism described below [57]. Effectively, the number of attention weights in this attention scheme are reduced to $N(H+W)$ from NHW, which greatly helps in faster and better convergence.To begin with, the positional queries Q_p are added to the outputs of *Self Attention* to obtain the cross attention query inputs Q. The attention occurs in two stages. First, given the encoder representations $M \in H \times W \times C$, we perform global average pooling to obtain a reduced row feature map $M_x \in W \times C$. The attention weights are computed as usual:

$$\Psi_x = \text{softmax}\Big(\frac{QK_x^T}{\sqrt{d_k}}\Big), \quad \Psi_x \in N \times W$$

where $K_x = M_x + K_{p,x}$ with $K_{p,x}[i] = \text{MLP}(\text{PE}(i/W))$ where PE stands for positional encoding. The output of the first stage (Z_x) is then computed as: $Z_x = \texttt{WeightedSum}(\Psi_x, M), \quad Z_x \in N \times H \times C$. Here $Z_x[i] = \Psi_x[i] \odot M$ and $\odot$ is the Hadamard product. Z_x is used as the value to the second stage attention computation. We again perform global average pooling on M to obtain a reduced column feature map $M_y \in H \times C$. The attention weights become:

$$\Psi_y = \text{softmax}\Big(\frac{QK_y^T}{\sqrt{d_k}}\Big), \quad \Psi_y \in N \times H$$

where $K_y = M_y + K_{p,y}$ and $K_{p,y}[i] = \text{MLP}(\text{PE}(i/H))$. The final cross-attention outputs Q_{xy} are computed as follows: $Q_{xy} = \texttt{WeightedSum}(\Psi_y, Z_x), \quad Q_{xy} \in N \times C$. Here $Q_{xy}[i] = \Psi_y[i] \odot Z_x$.

Final Predictions: The line queries undergo successive layers of self-attention and cross-attention with the encoder representations. Ultimately, each query $Q^{(i)}$ is transformed into an output embedding $E^{(i)}$ - see Fig. 4.3. Each $E^{(i)}$ either represents a scribble or ϕ (the empty class). Three Feed Forward

Networks (FFNs) f_1, f_2 and f_3 are applied to $E^{(i)}$ to obtain the scribble line parameters μ_x, μ_y, m along with a line probability score p: $f_1(E) = (\mu_x, \mu_y)$; $f_2(E) = m$; $f_3(E) = p$. Note that we have dropped the index i for clarity. During inference, only line queries with a probability above a threshold p_0 are selected as final scribbles.

4.4.3 Text-Energy Map Generator

The text-energy map generator is used to generate a continuous binary ($[0, 1]$) map to be used as an input to the seam generation algorithm in *Stage-2*. It uses a hybrid CNN-transformer architecture (see Fig. 4.5). The input patch is fed to a CNN encoder which outputs a feature map P. This feature map and representation M from the backbone encoder are fed to a transformer decoder. Within the decoder, self attention is applied to M and the result is processed along with P via a standard cross attention mechanism [49]. The resulting output is decoded to the target binary map via a CNN decoder containing skip connections with intermediate feature maps of the CNN encoder.

4.5 Stage 1 : Inference & Post-Processing Algorithm

In this section, we describe the mechanism by which global document-level strike-through scribbles are obtained during test time (inference). The proposed inference is elaborate to tackle multiple aspects like varying text-density across document collections, resolving corner-cases for scribble merge issues,etc.

Algorithm 3 Context-Adaptive Patching

▷ Input Manuscript Image $\mathcal{I}$
▷ Output A list of context adapted patches $\mathcal{P}$
$s \leftarrow$ Stage-1 model's input patch size
$\zeta \leftarrow$ integer scaling factor
$\pi \leftarrow [64 \times 64, 128 \times 128, \ldots 512 \times 512]$ ▷ candidate patch sizes
gaps $\leftarrow$ [] ▷ stores the median gap values per patch
for m in π **do** ▷ m: candidate patch size
    $\mathcal{P}_m \leftarrow$ SAMPLEPATCHES($\mathcal{I}, m$) ▷ sample overlapping $m \times m$ patches
    $\mathcal{P}_s \leftarrow$ RESIZEPATCHES($\mathcal{P}_m, s$) ▷ Resize to $s \times s$ patches
    $\mathcal{O}_m \leftarrow$ LINEPARAMETERGENERATOR($\mathcal{P}_s$) ▷ Sec. 4.4.2
    for o in $\mathcal{O}_m$ **do** ▷ o: line param outputs for a patch
        $\delta \leftarrow$ COMPUTEMEDIANINTERLINEGAP(o)
        $\delta \leftarrow \delta \cdot \frac{m}{s}$ ▷ normalize wrt candidate patch size
        gaps.append(δ)
    end for
end for
$t = \lceil$ MEDIAN(gaps) $\rceil \cdot \zeta$ ▷ the adaptive patch size ▷ $\lceil \cdot \rceil$: ceiling function
$\mathcal{P} \leftarrow$ SAMPLEPATCHES($\mathcal{I}, t$)
return $\mathcal{P}$

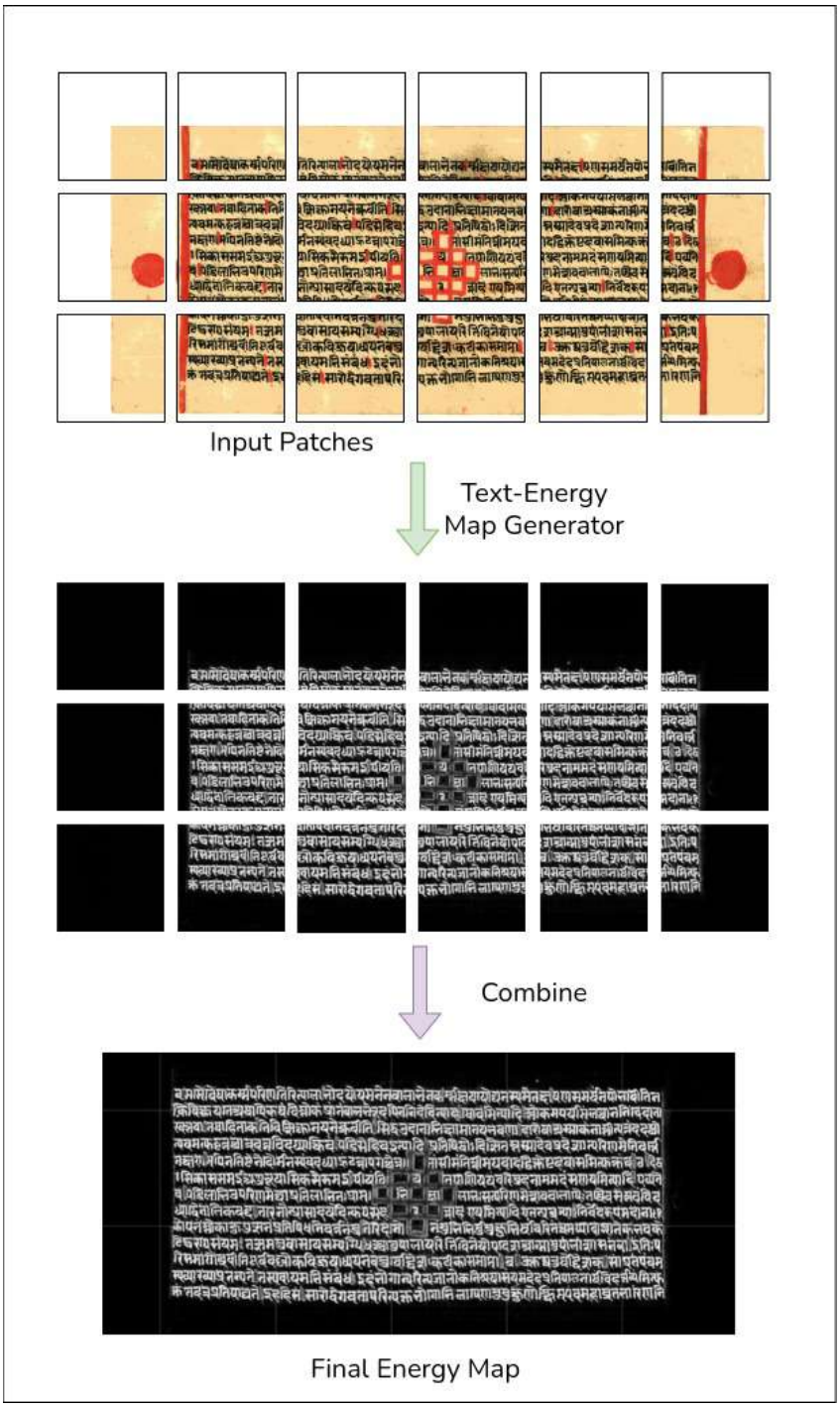


Figure 4.4: Combining patch-level Text-Energy outputs.

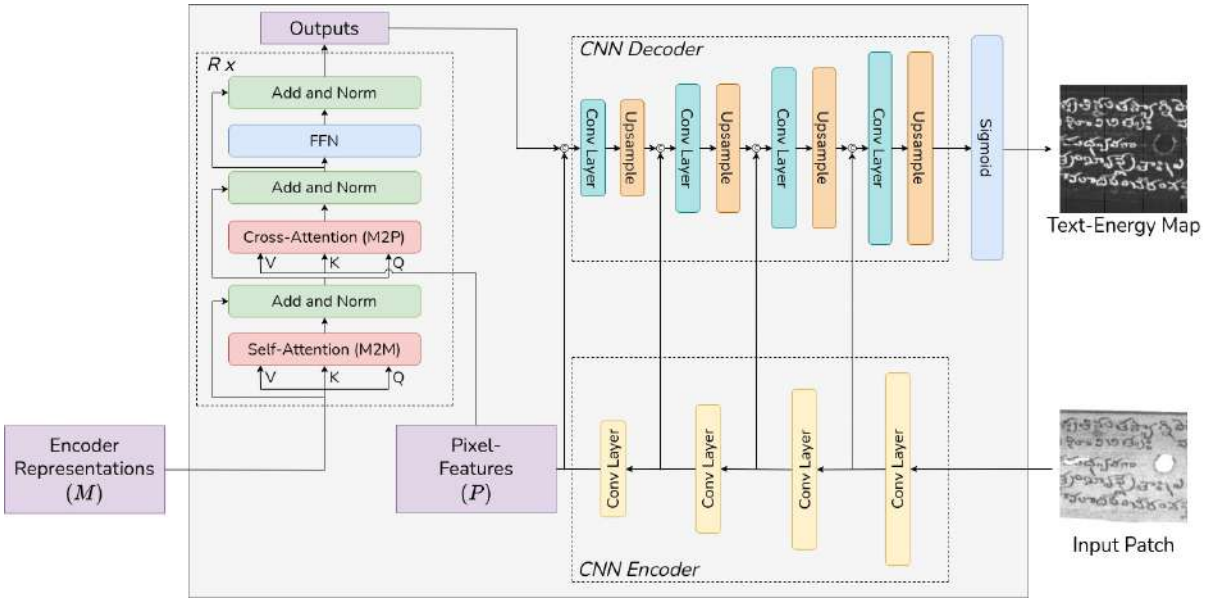


Figure 4.5: Text-Energy Map Generator

4.5.1 Context-Adaptive Patching

Given that historical datasets often occur in high dimensions and varying text densities. A fixed-size patching approach is not optimal, as it might not contain sufficient contextual information for effective scribble line prediction. When sampling across documents, observing fixed-sized patches shows that few patches might contain multiple lines (dense documents), and some might contain very few characters (sparse documents). To combat this issue, we try to sample the document from multiple patch scales and then feed it into the line parameter model to estimate the average spacing between text lines (refer Fig. 4.6. We call this patching mechanism as 'context' adaptive as it is more aware of the document under different settings (refer Algorithm **??**).

4.5.2 Projection-Merging

Having obtained the list of patches $\mathcal{P}$, we *project* their scribble line outputs onto the original image, and *merge* them appropriately to produce a global scribble map $\mathcal{S}$. This is done using an iterative algorithm, described in Algorithm 4. We iterate over the list of patches $\mathcal{P}$. At each step in the iteration, $\mathcal{S}$ contains information about the scribbles encountered till that step. All the points in $\mathcal{S}$ with pixel value i represent the i^{th} scribble (one-indexed). The basic idea is that each *new* line $\hat{l}$ in a patch can change the state of $\mathcal{S}$ by either (a) refining and/or extending an existing scribble, or (b) introducing a new scribble.Similar to this, we also combine the patch-level Text-Energy outputs to generate a global Text-Energy map $\mathcal{B}$ for the complete image.

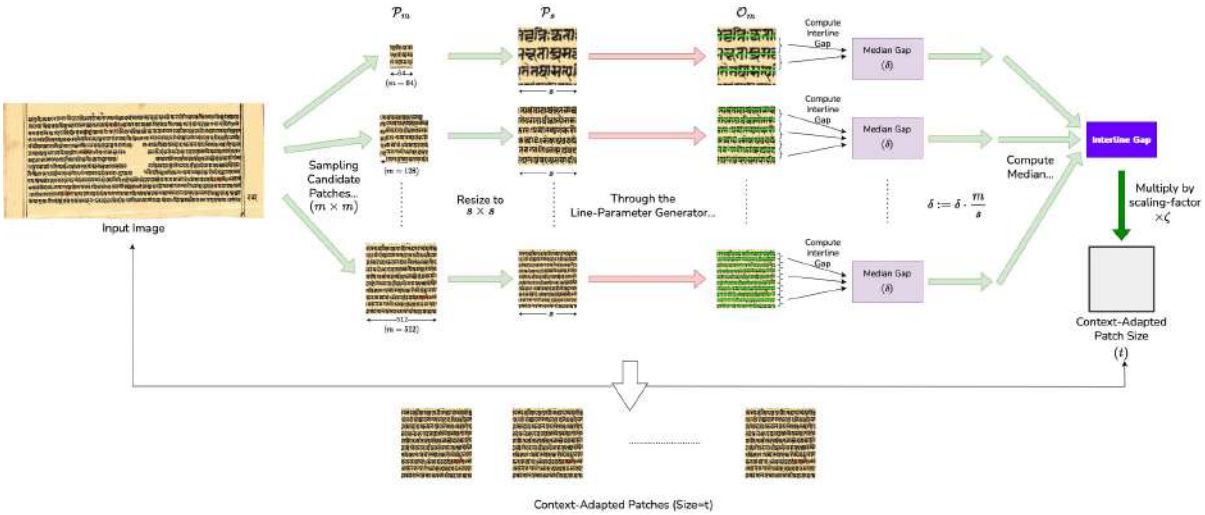


Figure 4.6: Context Aware Patching in Jain Manuscript of Indiscapes2 dataset [1]. More details of this patching mechanism can be seen in Algorithm 3.

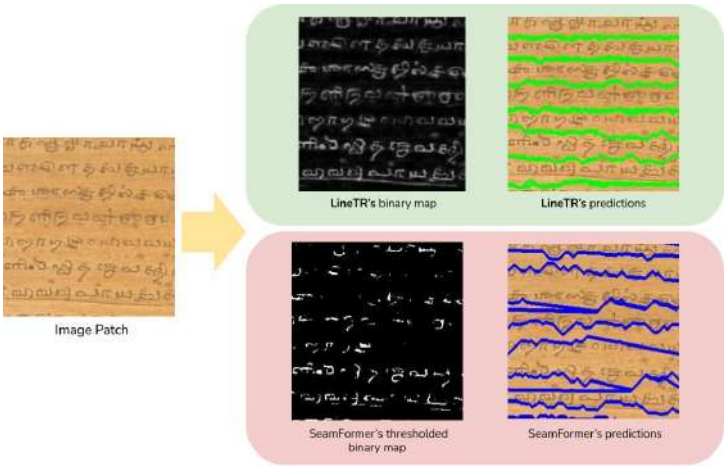


Figure 4.7: A comparision between the binary maps of SeamFormer[4] and **LineTR**.

4.6 Stage II

The outputs of *Stage-1* are a list of global scribbles and a *continuous* Text-Energy binary map $\mathcal{B}$ of the complete input image. These are processed by the seam generation pipeline from SeamFormer [4] to obtain tight fitting polygons enclosing the text lines. We introduce two crucial modifications to the default approach in SeamFormer [4]. One, instead of using thresholded binary map, we use the output from *Stage-1* as it is, i.e. $\mathcal{B}$ contains floating point values in the range $[0, 1]$. This avoids loss of crucial text presence information caused by thresholding as show in Fig.(4.7). The second modification is to discard other energy maps used in SeamFormer [4] (smoothing map, diacritic map and sobel map). Doing so eliminates the need for determining energy map weight coefficients.

Algorithm 4 Projection-Merging Algorithm

▷ **Input** The list of patches $\mathcal{P}$
▷ **Output** The global scribble map $\mathcal{S}$
$H, W \leftarrow$ height, width of the input image $\mathcal{I}$
$\mathcal{S} \leftarrow$ empty image of shape $H \times W$ initialized with zeros ▷ the global scribble map
$\eta \leftarrow 0$ ▷ the number of scribbles (globally)
for p in $\mathcal{P}$ **do** ▷ for each patch $p \in \mathcal{P}$
    $\omega \leftarrow$ region bounded by p in the original image ▷ $\omega \in \mathbb{R}^4$, xyxy coordinates
    $W \leftarrow \mathcal{S}[\omega]$ ▷ cropped section ω of $\mathcal{S}$
    $o \leftarrow$ LINEPARAMETERGENERATOR(p) ▷ outputs for patch p
    for l in o **do** ▷ l is a predicted line in o
        $\mathcal{L} \leftarrow \{(x, y) \mid (x, y) \in W,\ m(x - \mu_x) + (y - \mu_y) = 0\}$ ▷ the *new* points
        newline $\leftarrow$ True ▷ False if l is used to extend an existing scribble
        for ν in $\{1, 2, \ldots, \eta\}$ **do** ▷ every possible pixel value in W
            $\mathcal{X}_\nu \leftarrow \{(x, y) | \ (x, y) \in W,$ s.t. $W[x, y] = \nu\}$ ▷ the *existing* points
            $\mathcal{D} \leftarrow []$ ▷ stores shortest distance of each point in $\mathcal{L}$ to some point in $\mathcal{X}_\nu$
            for (x, y) in $\mathcal{L}$ **do** ▷ for all the *new* points
                $d \leftarrow$ MINDIST((x, y), $\mathcal{X}_\nu$) ▷ min distance of (x, y) to a point in $\mathcal{X}_\nu$
                if $d \leq \kappa\delta$ **then** ▷ if the *new* point is close to some *existing* point
                    $\mathcal{D}$.append(d)
                end if
            end for
            if $\mathcal{D}$.size $\geq \tau|\mathcal{L}|$ **then** ▷ refine/extend the existing scribble
                $\mathcal{L}_{\text{pca}} \leftarrow$ COMBINEPOINTS($\mathcal{X}_\nu$, $\mathcal{L}$) ▷ PCA of $\mathcal{X}_\nu \cup \mathcal{L}$
                newline $\leftarrow$ False
                SETPIXELSTOVALUE(W, $\mathcal{X}_\nu$, 0) ▷ sets pixel locations $\mathcal{X}_\nu$ in W to 0
                SETPIXELSTOVALUE(W, $\mathcal{L}_{\text{pca}}$, ν)
            end if
        end for
        if newline is True **then** ▷ l forms a new scribble in global scribble map $\mathcal{S}$
            SETPIXELSTOVALUE(W, $\mathcal{L}$, $\eta + 1$)
            $\eta \leftarrow \eta + 1$
        end if
    end for
    $\mathcal{S}[\omega] \leftarrow W$ ▷ update $\mathcal{S}$
end for

4.7 Loss Functions

In Line Parameter Prediction module , having obtained the predicted parameters and line probability scores for each query, we match each ground truth line to a query such that the assignment is one-to-one and optimal. This is done by scoring each ground truth line against every prediction using a cost function $\mathcal{C}_{\text{match}}$. The optimal assignment is computed using the Hungarian algorithm [59].

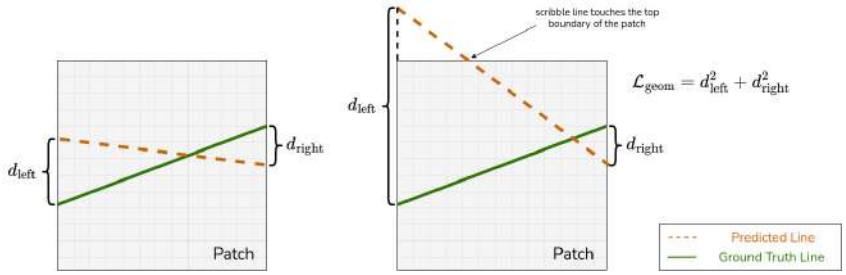


Figure 4.8: Our proposed loss function for penalizing the misalignment between two lines. If the scribble line touches the top or the bottom patch boundary, then we use interpolation to calculate the loss as shown in the figure on the right.

Let $\{l_i\}_{i=1}^{N_0}$ be the set of ground truth lines sorted by their μ_y values. We define the median gap δ between the sorted lines as $\delta = \texttt{median}\{\mu_{y_2} - \mu_{y_1}, \mu_{y_3} - \mu_{y_2}, ..., \mu_{y_{N_0}} - \mu_{y_{N_0-1}}\}$. Let $\hat{l}(\hat{\mu_x}, \hat{\mu_y}, \hat{m})$ be a predicted line with associated probability p and $l(\mu_x, \mu_y, m)$ be a ground truth line. We propose a geometry-based loss function for penalizing the misalignment between the predicted line $\hat{l}$ and the ground truth line l. Let d_{left} and d_{right} be the vertical distances between the lines at the left and the right patch boundary respectively (see Fig. 4.8). We define the geometric loss $\mathcal{L}_{\text{geom}}$ as follows:

$$\mathcal{L}_{\text{geom}}(\hat{l}, l) = d^2_{\text{left}} + d^2_{\text{right}}$$

Finally, we define the matching cost function $\mathcal{C}_{\text{match}}$ as follows:

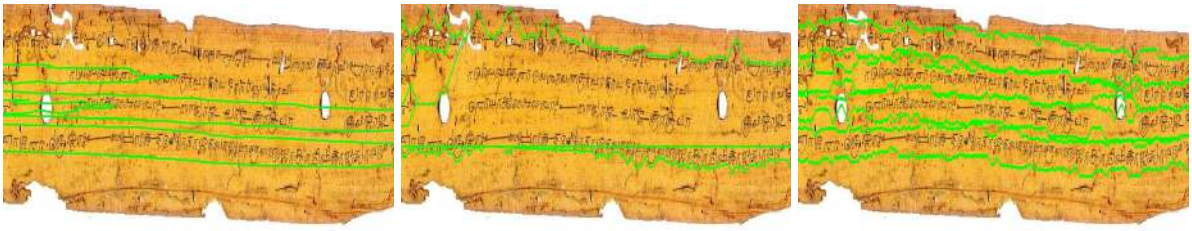
$$\mathcal{C}_{\text{match}}(\hat{l}, l) = \lambda_{\text{geom}} \cdot \frac{\mathcal{L}_{\text{geom}}(\hat{l}, l)}{\delta} - \lambda_p \cdot p$$

where $\lambda_{\text{geom}}, \lambda_p \in \mathbb{R}$ are hyperparameters. After the matching, some queries would be assigned to a line. The rest of the queries would be assigned ϕ (the empty class). Let $g : \mathbb{Z}^+ \to \{\mathbb{Z}^+, \phi\}$ be the obtained matching, such that $g(i) = j$ if the i^{th} prediction is matched to the j^{th} ground truth line, and $g(i) = \phi$ if it is assigned ϕ. We define the optimization loss $\mathcal{L}_{\text{opt}}$ as follows:

$$\begin{aligned}
\mathcal{L}_{\text{opt}} &= \mathcal{L}_1 + \mathcal{L}_2 \\
\mathcal{L}_1 &= \sum_{i=1; g(i)=\phi}^{N} \left[\lambda_1 \mathcal{L}_{\text{focal}}(p_i, -) \right] \\
\mathcal{L}_2 &= \sum_{i=1; g(i)\neq\phi}^{N} \left[\lambda_2 \mathcal{L}_{\text{focal}}(p_i, +) + \lambda_{\text{geom}} \frac{\mathcal{L}_{\text{geom}}(\hat{l}, l)}{\delta^2} \right]
\end{aligned}$$

where $\lambda_{\text{geom}}, \lambda_1, \lambda_2 \in \mathbb{R}$ are hyperparameters, and $\mathcal{L}_{\text{focal}}$ stands for the focal loss [60]:

$$\begin{aligned}
\mathcal{L}_{\text{focal}}(p_i, +) &= -(1 - p_i)^{\gamma} \log(p_i) \\
\mathcal{L}_{\text{focal}}(p_i, -) &= -(p_i)^{\gamma} \log(1 - p_i)
\end{aligned}$$



(a) Palmira (b) SeamFormer (c) **LineTR**

Figure 4.9: Performance comparison on a challenging image with curved text-lines

where γ is a hyperparameter.
Now for the optimisation of Text-Energy Map generator branch, we use the focal loss [60]:

$$\mathcal{L} = -\alpha \cdot y \cdot (1\hat{y})^{\gamma} \log \hat{y} - (1-\alpha) \cdot (1y) \cdot \hat{y}^{\gamma} \log(1\hat{y})$$

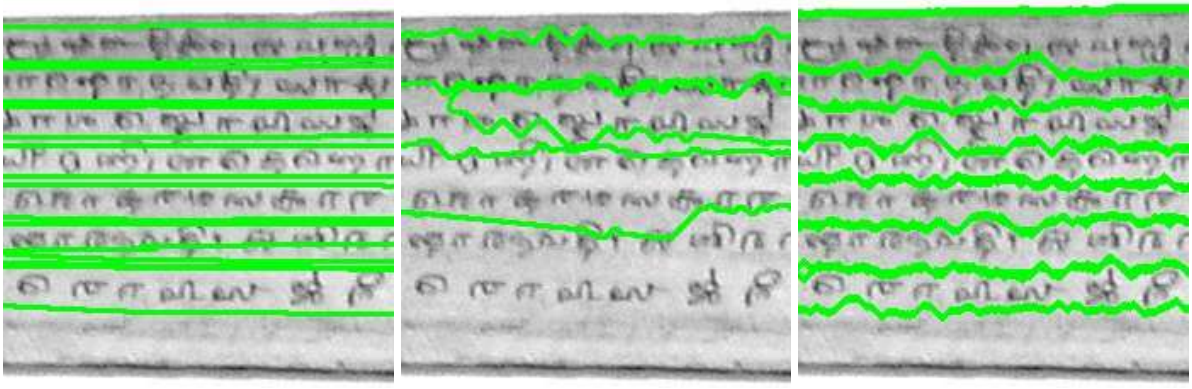
where $\hat{y}$ represents the sigmoid activation layer's output (shaded blue in Fig. 4.5) , $y \in \{0, 1\}$ represents the ground truth, and α, γ are hyperparameters. This section concludes various loss functions that have been used across modules.

4.8 Training Details

We perform *Stage-1* training in two phases. In the first phase, we train only the ViT backbone and the Line-Parameter Generator. In the second phase, we freeze both of them, and train only the Text-Energy Map Generator. In the first phase, we use a learning rate of 5×10^{-5}, and train for 60 epochs. We then reduce the learning rate to 10^{-5} and train for another 20 epochs. In the second phase, we train only the energy map generator with a learning rate of 10^{-4} for 50 epochs. We use the AdamW optimizer, with the coefficients set to PyTorch's defaults. Our implementation is based on the distributed PyTorch Lightning framework. We trained our model on 4 NVIDIA RTX 2080Ti GPUs, with 24 images on each GPU. The first phase of training took around 32 hours, and the second phase took around 7 hours.

4.9 Results

As Table 4.1 shows, **LineTR** clearly outperforms other models by a significant margin across all datasets. The consistently poor scores among other baselines is an outcome of loose fit predicted text regions, often missing crucial textual elements such as diacritics. Many baseline methods [38, 1, 35], approach text line segmentation as a per-pixel classification task, resizing images with large aspect ratios to a fixed lower size. This resizing tends to merge adjacent predicted text, particularly in dense text documents. In other baselines [4, 30], the suboptimal results are due to excessive dataset-specific decisions. **LineTR**'s numbers on unseen datasets are on par with ones encountered during training. This shows its zero-shot generalization ability.



(a) Palmira (b) SeamFormer (c) **LineTR**

Figure 4.10: Performance comparison on a challenging image with very dense text

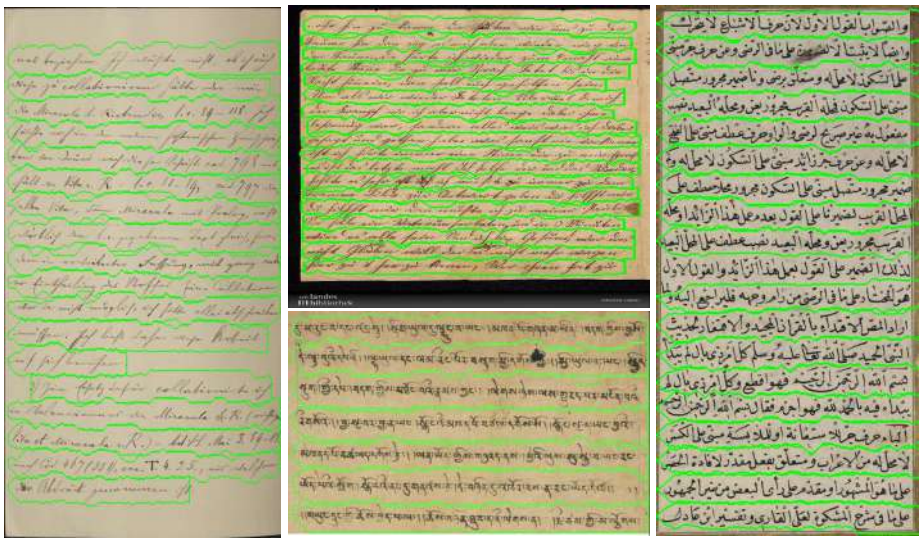


Figure 4.11: Qualitative results of LineTR on unseen datasets. In clockwise order, we have two English handwritten images picked from ICDAR 2017 competition, a challenging Urdu manuscript followed by Tibetan manuscript.

Table 4.1: Comparative evaluation of **LineTR** on benchmark datasets (Sec. 4.9).

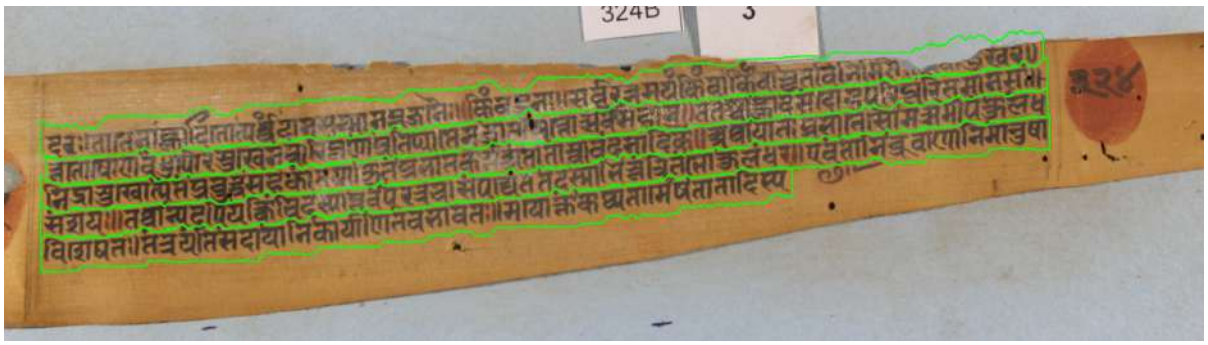
	I2[1]	SD[54]	BL[53]	KH[55]	KG[4]	WM*	UB*	SM*
				AvgHD ↓				
Doc-UFCN [35]	68.60	71.17	74.52	95.90	38.79	41.39	71.10	174.14
SeamFormer [4]	11.82	8.92	104.75	49.03	7.83	11.54	9.46	130.98
Palmira [1]	15.81	6.50	301.21	203.59	7.50	24.11	18.88	15.78
LCG [30]	16.82	39.65	95.18	44.50	29.72	317.98	481.14	1034.88
docExtractor [38]	77.25	33.86	43.16	48.36	40.24	87.75	95.51	260.70
LineTR	**1.86**	**1.30**	**22.62**	**14.97**	**2.09**	**0.94**	**1.01**	**3.04**
				IoU ↑				
Doc-UFCN [35]	0.23	0.10	0.08	0.11	0.12	0.15	0.10	0.16
SeamFormer [4]	0.51	0.53	0.32	0.37	0.49	0.49	0.76	0.43
Palmira [1]	0.72	0.66	0.39	0.41	0.62	0.53	0.54	0.69
LCG [30]	0.37	0.12	0.12	0.18	0.20	0.02	0.01	0.07
docExtractor [38]	0.03	0.01	0.00	0.02	0.12	0.03	0.03	0.02
LineTR	**0.80**	**0.73**	**0.62**	**0.69**	**0.81**	**0.66**	**0.82**	**0.82**

* Newly introduced datasets with zero-shot baseline testing.

4.10 Conclusion

LineTR is a novel dataset-agnostic approach for robust text line segmentation in diverse and challenging historical manuscripts. Similar to recent successful approaches, we use a two stage approach - scribble generation and scribble-conditioned polygon generation. However, our unique and novel design choices make a significant difference. The choice of predicting per-patch scribble line parameters in the first stage helps avoid the difficulties of pixel-based scribble representation. Our adaptive patch extraction ensures sufficient context capture for predicting scribble line parameters. A sensible design for Text-Energy Map Generator not only simplifies second stage processing, it also improves overall results. Unlike existing approaches, **LineTR**'s methodology does not require dataset-specific fine-tuning. Another distinction is that the training process results in a single model and does not require dataset-specific models. These features make **LineTR** advantageous from a maintainability and scalability point of view. **LineTR** not only outperforms strong baselines but also exhibits good zero-shot performance on unseen datasets. This showcases its generalizability and utility for the community.

(a) WM



(b) UB

(c) SM

Figure 4.12: Zero-shot outputs of **LineTR** on the newly introduced datasets.

Chapter 5

Conclusion & Future Works

5.0.1 Conclusion

This thesis concentrates on the precise segmentation of text lines within unstructured handwritten documents. It addresses the challenges inherent in this domain and the specific task at hand. A comprehensive examination of the difficulties encountered in historical handwritten documents is provided, along with an analysis of the constraints associated with achieving precise text-line segmentation in these documents. This thesis presents SeamFormer, a multi-task Vision Transformer model that tackles text-line segmentation. We discuss how our design choices help with extreme aspect ratios and highly dense document images. Unlike existing approaches, SeamFormer effectively processes variable-sized images by dividing them into fixed-size patches. SeamFormer performs the relatively simple task of detecting a modified version of text baselines (namely scribbles) to better locate the text region. It also proposes a scribble-conditioned seam generation technique that enables text line delineation based on each generated scribble. The energy maps proposed for this algorithm also aim to introduce diacritic awareness. As an improvement to SeamFormer, we propose LineTR, a multi-task DETR-based model that exhibits superior performance via zero-shot operation on unseen documents. Instead of generating coarse text blobs and solving it as a binary classification problem, LineTR approaches it as a geometrical structure prediction problem. LineTR also performs context-aware patching of the original image as opposed to fixed-size patching, resulting in better text line accuracy. This model features a single energy map repurposed from binary map prediction. Furthermore, we highlight the adaptation of HINDOLA [5], an annotation tool, through the incorporation of Stage II of SeamFormer, supplemented by the inclusion of scribble functionality and a pre-generated collection of binarized images. This augmentation sped up the annotation process and necessitated minimal corrections during the final post-processing phase. Given all these developments in multiple aspects of the project, here are a few potential future work directions that can be suggested.

5.0.2 Future Works

A notable accomplishment of LineTR was its ability to yield robust zero-shot results, offering promising prospects for the development of a large-scale universal model targeting handwritten documents. For universality of a model , the large scale dataset should be diverse with multiple text-samples

with varying backgrounds,languages,density, orientation etc for accurate scribble detection. To achieve this, it is necessary to aggregate multiple handwritten datasets sourced from diverse academic and publicly available repositories. Notably, emphasis should be placed on encompassing linguistic diversity to capture the grammatical intricacies specific to each language. In terms of architecture, we should try adopt more complex models beyond DETR that can produce superior quality of scribbles by modelling n-dimensional curves. This action results in abandoning the assumption that we previously relied on, which assumed that the scribble within a certain image patch could be represented as a straight line with a corresponding slope. Our future direction of work must be aligned to exploring n-dimensional polynomial curves such as smooth Bezier curve. Additionally, the Stage-II aspect of the model be engineered to produce flexible seams exploring more direction whilst maintaining the precision of the final text polygons generated. Upon successful implementation, this approach can be seamlessly integrated into the fully automated mode of the existing HINDOLA [5] tool. Such integration would serve to significantly reduce the overall time required for polygonal annotation in new datasets.

Another school of thought , address the text line annotation challenge, we adopt Reinforcement Learning (RL) approaches for mimicking human annotation behaviour. There have been similar works in the domain of AI x Art - specifically in the domain of painting , where RL agents have been trained to learn aspects like thickness , colors and position of a paint brush on a canvas. These approaches are often based on Behavioural Cloning (BC) and Inverse Reinforcement Learning(IRL) based RL models. We can take inspiration from such existing models and try to adapt them to simple annotation procedures. The ultimate goal is to design data-agnostic, lightweight models that deliver superior performance across large-scale handwritten datasets. These models aim to simplify the annotation process, facilitating the creation of more data which can then be developed and further annotated for OCR pipelines.

Appendix A

Appendix

A.1 HInDoLa-SeamFormer Annotation Tool

As highlighted in Chapter 4 4 , one of the major contribution was the introduction of SouthEast Asian descent documents to the existing large scale Indiscapes2 [1] - UB,WM,TM. These datasets and annotated polygons from scratch via a modified setup of HINDOLA [5]. We utilize the accuracy facilitated by Stage-II to circumvent the need for annotators to extensively annotate polygons , which becomes tedious in highly dense text collection. The HINDOLA [5] tool, has multiple outlining utilities for the user to annotation like polyline, free-hand and polygon. To this we add another option - known as 'scribble'. To illustrate , instead of annotating the whole document image , the user is only instructed to draw scribbles that pass through the text line (refer Fig. A.2).

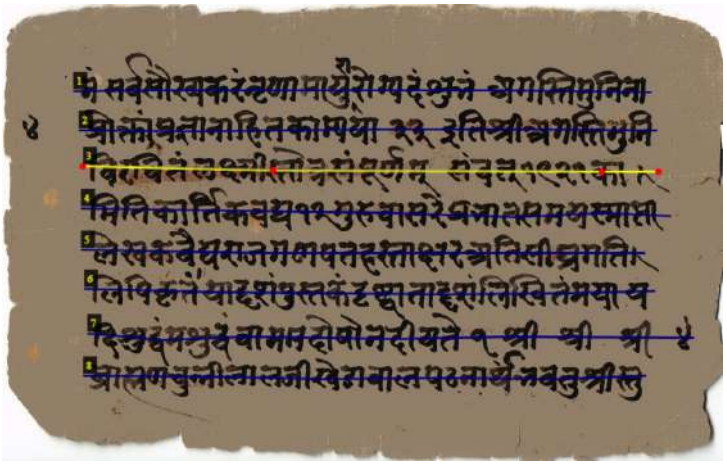


Figure A.1: Scribble Annotation on a sample Indic document(Sec. A.1)

On the backend of the tool, the annotations pertaining to these scribbles are resampled based to get uniformly sampled points. The user input is combined with the binarised image outputs generated by Stage-I SeamFormer [4].These are then combined and supplied to Stage-II of SeamFormer to generate polygonal outputs.

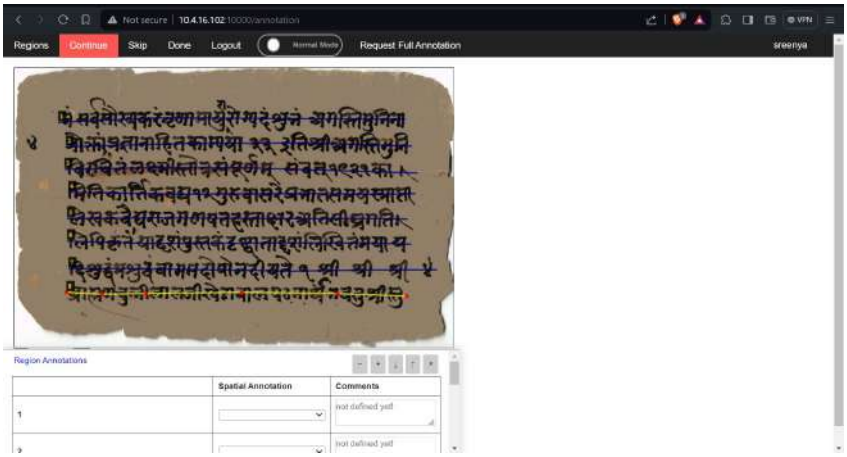


Figure A.2: HINDOLA [5] Annotation System frontend with the capabilities of labelling and drawing a scribble. (Sec. A.1)

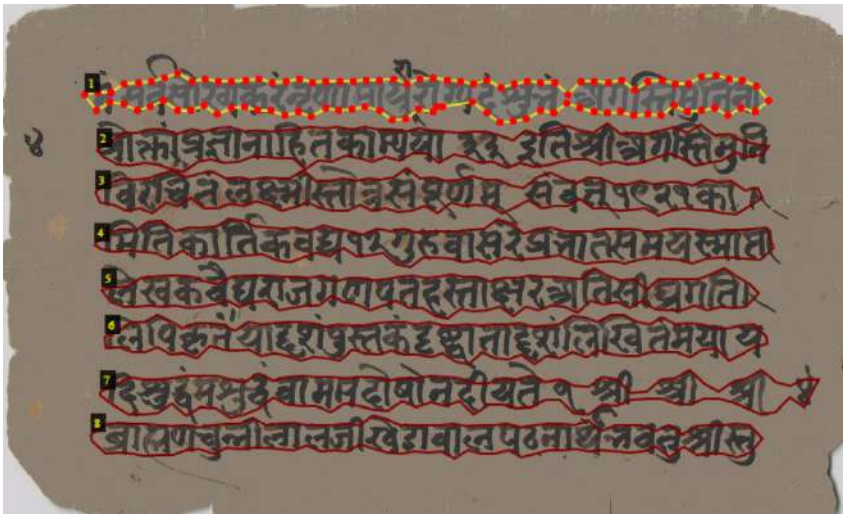


Figure A.3: Output of scribble annotations processed by SeamFormer-Hindola annotator pipeline(Sec. A.1)

In Fig. A.3, you can observe that the annotator only needed to make a few corrections, adding some diacritics and characters. The accuracy of the text lines and the simplicity of the annotation process are the key advantages of this annotation method. With this approach, 50 images of UB, WM & TM collections were successfully annotated.

Publications

1. (ICDAR 2023) **SeamFormer** : High Precision Text Line Segmentation for Handwritten Documents. Authors - **Niharika Vadlamudi** , Rahul Krishna & Ravi Kiran Sarvadevabhatla.

2. (ICPR 2024) **LineTR:** Unified Text Line Segmentation for Challenging Palm Leaf Manuscripts. Authors - Vaibhav Agrawal, **Niharika Vadlamudi**, Muhammad Waseem , Amal Joseph , Sreenya Chitluri & Ravi Kiran Sarvadevabhatla.

Bibliography

[1] S. Sharan, S. Aitha, A. Kumar, A. Trivedi, A. Augustine, and R. K. Sarvadevabhatla, "Palmira: a deep deformable network for instance segmentation of dense and uneven layouts in handwritten manuscripts," in *International Conference on Document Analysis and Recognition*, pp. 477–491, Springer, 2021. ix, x, xi, 2, 4, 7, 8, 14, 17, 18, 19, 29, 30, 31, 38, 41, 43, 47

[2] A. Prusty, S. Aitha, A. Trivedi, and R. K. Sarvadevabhatla, "Indiscapes: Instance segmentation networks for layout parsing of historical indic manuscripts," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pp. 999–1006, IEEE, 2019. ix, 2, 8, 10, 14

[3] M. W. A. Kesiman, D. Valy, J. C. Burie, E. Paulus, M. Suryani, S. Hadi, M. Verleysen, S. Chhun, and J.-M. Ogier, "Icfhr 2018 competition on document image analysis tasks for southeast asian palm leaf manuscripts," in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp. 483–488, 2018. xi, 27, 30

[4] N. Vadlamudi, R. Krishna, and R. K. Sarvadevabhatla, "Seamformer: High precision text line segmentation for handwritten documents," in *International Conference on Document Analysis and Recognition*, pp. 313–331, Springer, 2023. xi, 17, 19, 31, 32, 38, 41, 43, 47

[5] A. Trivedi and R. K. Sarvadevabhatla, "Hindola: a unified cloud-based platform for annotation, visualization and machine learning-based layout analysis of historical manuscripts," in *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, vol. 2, pp. 31–35, IEEE, 2019. xi, 17, 32, 45, 46, 47, 48

[6] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017. 2, 7, 8

[7] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125, 2017. 2

[8] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pp. 740–755, Springer, 2014. 3

[9] D. Hoiem, S. K. Divvala, and J. H. Hays, "Pascal voc 2008 challenge," *World Literature Today*, vol. 24, no. 1, pp. 1–4, 2009. 3

[10] A. Alaei, U. Pal, and P. Nagabhushan, "A new scheme for unconstrained handwritten text-line segmentation," *Pattern Recognition*, vol. 44, no. 4, pp. 917–928, 2011. 6

[11] L. O'Gorman, "The document spectrum for page layout analysis," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 15, no. 11, pp. 1162–1173, 1993. 6

[12] Y. Pu and Z. Shi, "A natural learning algorithm based on hough transform for text lines extraction in handwritten documents," in *Advances in Handwriting Recognition*, pp. 141–150, World Scientific, 1999. 6

[13] L. Likforman-Sulem and C. Faure, "Extracting text lines in handwritten documents by perceptual grouping," *Advances in handwriting and drawing: a multidisciplinary approach*, pp. 117–135, 1994. 6

[14] E. Bruzzone and M. C. Coffetti, "An algorithm for extracting cursive text lines," in *Proceedings of the Fifth International Conference on Document Analysis and Recognition. ICDAR'99 (Cat. No. PR00318)*, pp. 749–752, IEEE, 1999. 6

[15] J. He and A. C. Downton, "User-assisted archive document image analysis for digital library construction," in *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.*, pp. 498–502, IEEE, 2003. 6

[16] G. Nagy, S. C. Seth, and S. D. Stoddard, "Document analysis with an expert system," in *Pattern recognition in practice II*, pp. 149–155, 1985. 6

[17] T. Pavildas, "Page segmentation by white streams," in *Proc. 1st Int. Conf. Document Analysis and Recognition*, pp. 945–953, 1991. 6

[18] A. Zahour, B. Taconet, P. Mercy, and S. Ramdane, "Arabic hand-written text-line extraction," in *Proceedings of Sixth International Conference on Document Analysis and Recognition*, pp. 281–285, IEEE, 2001. 6

[19] Z. Shi, S. Setlur, and V. Govindaraju, "Text extraction from gray scale historical document images using adaptive local connectivity map," in *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)*, pp. 794–798, IEEE, 2005. 6

[20] Z. Shi, S. Setlur, and V. Govindaraju, "A steerable directional local profile technique for extraction of handwritten arabic text lines," in *2009 10th International Conference on Document Analysis and Recognition*, pp. 176–180, IEEE, 2009. 6

[21] M. W. A. Kesiman *et al.*, "Benchmarking of document image analysis tasks for palm leaf manuscripts from southeast asia," *Journal of Imaging*, vol. 4, no. 2, p. 43, 2018. 7

[22] R. Chamchong and C. C. Fung, "Text line extraction using adaptive partial projection for palm leaf manuscripts from thailand," in *2012 International Conference on Frontiers in Handwriting Recognition*, pp. 588–593, IEEE, 2012. 7

[23] R. Chamchong and C. C. Fung, "Character segmentation from ancient palm leaf manuscripts in thailand," in *Proceedings of the 2011 Workshop on Historical Document Imaging and Processing*, pp. 140–145, 2011. 7

[24] D. Valy, M. Verleysen, and K. Sok, "Line segmentation for grayscale text images of khmer palm leaf manuscripts," in *2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA)*, pp. 1–6, IEEE, 2017. 7

[25] M. W. A. Kesiman, J.-C. Burie, and J.-M. Ogier, "A new scheme for text line and character segmentation from gray scale images of palm leaf manuscript," in *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp. 325–330, IEEE, 2016. 7

[26] S. Avidan and A. Shamir, "Seam carving for content-aware image resizing," in *ACM SIGGRAPH 2007 papers*, pp. 10–es, 2007. 7, 23, 25

[27] R. Saabni and J. El-Sana, "Language-independent text lines extraction using seam carving," in *2011 International Conference on Document Analysis and Recognition*, pp. 563–568, IEEE, 2011. 7, 25

[28] A. Asi, R. Saabni, and J. El-Sana, "Text line segmentation for gray scale historical document images," in *Proceedings of the 2011 workshop on historical document imaging and processing*, pp. 120–126, 2011. 7

[29] N. Arvanitopoulos and S. Süsstrunk, "Seam carving for text line extraction on color and grayscale historical manuscripts," in *2014 14th International Conference on Frontiers in Handwriting Recognition*, pp. 726–731, IEEE, 2014. 7

[30] M. Alberti, L. Vögtlin, V. Pondenkandath, M. Seuret, R. Ingold, and M. Liwicki, "Labeling, cutting, grouping: an efficient text line segmentation method for medieval manuscripts," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1200–1206, IEEE, 2019. 7, 8, 29, 41, 43

[31] D. Li, Y. Wu, and Y. Zhou, "Linecounter: Learning handwritten text line segmentation by counting," in *2021 IEEE International Conference on Image Processing (ICIP)*, pp. 929–933, IEEE, 2021. 7

[32] G. Renton, Y. Soullard, C. Chatelain, S. Adam, C. Kermorvant, and T. Paquet, "Fully convolutional network with dilated convolutions for handwritten text line segmentation," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 21, pp. 177–186, 2018. 7

[33] B. K. Barakat, A. Droby, R. Alaasam, B. Madi, I. Rabaev, R. Shammes, and J. El-Sana, "Unsupervised deep learning for text line segmentation," in *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 2304–2311, IEEE, 2021. 7

[34] B. Barakat, A. Droby, M. Kassis, and J. El-Sana, "Text line segmentation for challenging handwritten document images using fully convolutional network," in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp. 374–379, IEEE, 2018. 7

[35] M. Boillet, C. Kermorvant, and T. Paquet, "Multiple document datasets pre-training improves text line detection with deep neural networks," in *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 2134–2141, IEEE, 2021. 7, 29, 41, 43

[36] O. Mechi, M. Mehri, R. Ingold, and N. E. B. Amara, "Text line segmentation in historical document images using an adaptive u-net architecture," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pp. 369–374, IEEE, 2019. 7

[37] S. A. Oliveira, B. Seguin, and F. Kaplan, "dhsegment: A generic deep-learning approach for document segmentation," in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp. 7–12, IEEE, 2018. 7, 29

[38] T. Monnier and M. Aubry, "docExtractor: An off-the-shelf historical document element extraction," in *ICFHR*, 2020. 7, 29, 41, 43

[39] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015. 7

[40] A. Jindal and R. Ghosh, "Text line segmentation in indian ancient handwritten documents using faster r-cnn," *Multimedia Tools and Applications*, pp. 1–20, 2022. 7

[41] A. Prusty, S. Aitha, A. Trivedi, and R. K. Sarvadevabhatla, "Indiscapes: Instance segmentation networks for layout parsing of historical indic manuscripts," in *ICDAR*, pp. 999–1006, 2019. 7, 29

[42] T.-N. Nguyen, J.-C. Burie, T.-L. Le, and A.-V. Schweyer, "An effective method for text line segmentation in historical document images," in *2022 26th International Conference on Pattern Recognition (ICPR)*, pp. 1593–1599, IEEE, 2022. 8

[43] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *ICCV*, pp. 764–773, 2017. 11

[44] A. Trivedi and R. K. Sarvadevabhatla, "Boundarynet: An attentive deep network with fast marching distance maps for semi-automatic layout annotation," in *International Conference on Document Analysis Recognition, ICDAR 2021*, 2021. 14

[45] M. A. Souibgui, S. Biswas, S. K. Jemni, Y. Kessentini, A. Fornés, J. Lladós, and U. Pal, "Docentr: An end-to-end document image enhancement transformer," in *2022 26th International Conference on Pattern Recognition (ICPR)*, 2022. 19, 27

[46] I. Pratikakis, K. Zagoris, G. Barlas, and B. Gatos, "Icdar2017 competition on document image binarization (dibco 2017)," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 01, pp. 1395–1403, 2017. 19

[47] A. Garz, M. Seuret, F. Simistira, A. Fischer, and R. Ingold, "Creating ground truth for historical manuscripts with document graphs and scribbling interaction," in *DAS*, pp. 126–131, 2016. 19

[48] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *ICLR*, 2021. 21, 33

[49] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017. 21, 22, 34, 35

[50] J. Sauvola and M. Pietikäinen, "Adaptive document image binarization," *Pattern recognition*, vol. 33, no. 2, pp. 225–236, 2000. 27

[51] W. Niblack, *An introduction to digital image processing*. Strandberg Publishing Company, 1985. 27

[52] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: Fast and flexible image augmentations," *Information*, vol. 11, no. 2, 2020. 28

[53] M. W. A. Kesiman, J.-C. Burie, G. N. M. A. Wibawantara, I. M. G. Sunarya, and J.-M. Ogier, "Amadi_lontarset: the first handwritten balinese palm leaf manuscripts dataset," in *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp. 168–173, IEEE, 2016. 29, 43

[54] M. Suryani, E. Paulus, S. Hadi, U. A. Darsa, and J.-C. Burie, "The handwritten sundanese palm leaf manuscript dataset from 15th century," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1, pp. 796–800, IEEE, 2017. 29, 43

[55] D. Valy, M. Verleysen, S. Chhun, and J.-C. Burie, "A new khmer palm leaf manuscript dataset for document analysis and recognition: Sleukrith set," in *Proceedings of the 4th International Workshop on Historical Document Imaging and Processing*, pp. 1–6, 2017. 29, 43

[56] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*, pp. 213–229, Springer, 2020. 31, 33, 34

[57] Y. Wang, X. Zhang, T. Yang, and J. Sun, "Anchor detr: Query design for transformer-based detector," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 36, pp. 2567–2575, 2022. 34

[58] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable detr: Deformable transformers for end-to-end object detection," *arXiv preprint arXiv:2010.04159*, 2020. 34

[59] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955. 39

[60] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *ICCV*, pp. 2980–2988, 2017. 40, 41