Journal of Scientific and Technical Research
15 (2) 2025, 124-131
www.publishingindia.com/jstr
https://doi.org/10.21863/jstr/2025.15.2.014

Advanced Document Structure Recognition in Devanagari Script Using an Optimized YOLOv8 Pipeline

Shweta Singh[1*], Sudeep Varshney[2] and Ankur Choudhary[3]
[1]Department of Computer Science and Engineering, Sharda University, Greater Noida, Uttar Pradesh, India.
Email: singh.shweta2721@gmail.com
[2]Department of Computer Science and Engineering, Sharda University, Greater Noida, Uttar Pradesh, India.
Email: sudeep.varshney@sharda.ac.in
[3]Accenture Pvt. Ltd., Greater Noida, Uttar Pradesh, India. Email: ankur.tomer@gmail.com
[*]Corresponding Author

Abstract: Quality recognition of structural areas on printed documents is vital in other downstream processes like optical character recognition, document retrieval and digital archiving. Devanagari script has further problems that include the close morphology, headline writing and multi-tiered character constructions which often undermine the efficiency of generic document-analytic models. In this paper, Authors presented a deep learning model that is founded on YOLOv8 to identify structural elements in Devanagari printed texts. The method is trained and tested on PubLayNet dataset which is a large scale benchmark that consists of a variety of document structures and later adjusted to Devanagari script using targeted fine-tuning and region-specific annotations. The model is very precise in detecting fundamental structural components of text block, titles, table, lists, and figures and it shows high generalization in complicated pages of Devanagari. The experimental findings prove that YOLOv8 is an appropriate solution to detect document-structure quickly and accurately and provide a solid basis of larger Indic-language document processing pipelines. The structure helps to achieve high efficiency in the process of document digitization and structural preservation that is needed to support high-quality script-specific OCR systems.

Keywords: Deep learning, Devanagari script, Document structure detection, Object detection, PubLayNet, YOLOv8.

I. Introduction

Analysis of document structure is also the basis of digital archiving, automated text comprehension, and script-sensitive optical character recognition (OCR). The growing variety of document appearance, along with the high rate of growth in multilingual content in online libraries, has enhanced the necessity to establish powerful structural-region identification procedures that would effectively accommodate not only traditional variability but also script-related variability. Initial studies in this field were mainly based on texture descriptors, connected component features and segmentation based on rules. As an example, Vidyarthi *et al.* [1] used texture cues with linked components to distinguish between text and non-text areas in documents, whereas Namboodiri and Jain [2] offered one of the first thorough discussions of document structure and layout analysis with proposed definition of the key issues related to complex document structures.

As massive digitalization projects have emerged, structural understanding automated techniques have become more significant. Methods that work directly in compressed space were proposed in order to cut back on computational costs, including the correlation-entropy feature extraction by Javed *et al.* [3]. Later surveys, such as the prominent article by Binmakhashen and Mahmoud [4], pointed at the development of the heuristic-based algorithms into machine learning and deep learning schemes. More recent frameworks, such as HDPA proposed by Lenc *et al.* [5], showed the promise of a single architecture to process historical documents, which is a larger-scale evolution of end-to-end automated frameworks. Deep learning has largely contributed to the development of document structural analysis by allowing models to learn complicated visual features directly out of data. The domain-adaptive learning methods, including those suggested by Mishra [6], enhanced the cross-domain generalization by aligning the activity of classifiers in various document types. Layout extraction has also been investigated by neural network based methods, including the one by Gorai and Nene [7] that was used to identify text and regions by using supervised learning to extract the information used in layout extraction, as in the case of Gorai2020 [8]. A second category of ensemble based methods has also arisen such as a voting based mixture of deep network layout analysis results as introduced by Fateh

et al. [9]. In addition to layout detection, ancillary document analysis problems have also been reinforced with architectures like Mask R-CNN and U-Net, such as historical text-line segmentation, which is shown by Fizaine *et al.* [10], among others, to be stronger with these architectures and methods compared to standard methods.

Structural-region detection in Devanagari printed documents has not been studied extensively in spite of these developments. The unique features of the script like the running headline (shirorekha), intricate ligatures and compactly stacked glyphs tend to impair the functionality of generic models. This increasing amount of Devanagari material in administrative, educational, and archival sources supports the idea that having a high-precision and deep learning-based system capable of finding structures is necessary.

TABLE I: SUMMARY OF SELECTED DOCUMENT LAYOUT DETECTION STUDIES

Sr. No.	*Study*	*Key Contribution*
1.	Qi *et al.* [23]	YOLO-DLA framework for multi-layout detection.
2.	Deng *et al.* [24]	Enhanced YOLO modules for document precision.
3.	Wu *et al.* [25]	YOLOv8 with PCA-guided augmentation.
4.	Akanda [26]	Comparative study of DL de-tectors
5.	Zhang *et al.* [27]	WeLayout: ICDAR-winning layout segmentation system.
6.	Fizaine *et al.* [28]	Mask-RCNN vs U-Net for text-line extraction.
7.	Aktar *et al.* [29]	YOLOv11 for Bangla script Detection.
8.	Kumar and Lehal [13]	YOLOv8 for Punjabi newspaper layout detection.
9.	Anand *et al.* [18]	RanLayNet dataset for layout generalization.
10.	Kudale *et al.* [30]	Weakly supervised multilingual text detection

Recent object detectors, especially ones based on the YOLO family, have good prospects of document-structure understanding because they can have very high inference speeds and can recognize multi-scale objects in complex layouts. The use of these developments on large scale data collections like PubLayNet presents an avenue to creating more generalized but script sensitive systems. The analysis of the problems of the Devanagari document interpretation is conducted with the help of the framework with YOLOv8 trained on PubLayNet and adjusted to structural peculiarities of the script. The methodology will be able to produce an accurate and consistent detection of core document regions and, thus, enhance the reliability of downstream OCR and document understanding processes.

The rest of the work is structured in the following way. Section II provides the elaborated literature review that shows the development of relevant techniques and gaps existing. Section III outlines the plan methodology encompassing the system architecture and processing pipeline. Section IV describes the experimentation environment as well as the evaluation metrics in which performance is measured. Section V presents and discusses the findings that were obtained in the proposed framework. Section VI provides some concluding statements made by the overall findings, and Section VII includes some suggestions about the possible directions that future investigations can follow.

II. LITERATURE REVIEW

The focus of recent development of document structure analysis has been on deep learning, object detection systems, and multimodal feature extraction. Li *et al.* [11] showed that layout detection generates better understanding of printed documents through the pipelines of OCR, making the proper structural-region localization. In this vein, Santos Junior *et al.* [12] went further to offer a comparative analysis of the following versions of YOLOv8, YOLOv9, YOLOv10, and YOLOv11 and discovered that the more recent YOLO variants offered superior features of multi-scale detection, which is appropriate in complex document images. Research on Indic scripts has also advanced, with Kumar and Lehal [13] applying YOLOv8 to Punjabi newspapers and demonstrating strong generalization across script-dense documents. Broader document-processing insights were presented by Lenc *et al.* [14] through the HDPA framework, which highlighted the value of unified architectures for historical and degraded documents.

Domain adaptation strategies have been explored by Mishra [15], who introduced a classifier-alignment mechanism to mitigate cross-domain variations in layout detection tasks. Earlier neural-network-based approaches, such as the method proposed by Gorai and Nene [16], validated the feasibility of learning-based structural extraction but lacked the multi-scale precision of modern detection models.

Ensemble-driven approaches also contributed to layout analysis, with Fateh *et al.* [17] combining multiple deep networks through a voting-based mechanism to improve prediction consistency. Dataset-centric advancements have been made through RanLayNet by Anand *et al.* [18], which

supports domain adaptation and improves generalizability across document types. Multimodal understanding has further enriched structural analysis, as shown by Tu *et al.* [19], who introduced the LayoutMask architecture to enhance interactions between textual and spatial features. Additionally, Sun *et al.* [20] proposed PP-DocLayout, a unified model capable of scaling across heterogeneous document formats, reflecting the increasing trend toward flexible and generalizable deep learning systems. Collectively, these studies demonstrate a clear progression from classical feature-engineered methods to advanced deep learning pipelines that leverage multi- scale detection, domain adaptation, and multimodal reasoning for accurate document-structure understanding. A Devanagari character–encoded mix-merge vision transformer has been introduced for robust document layout analysis [21]. A recent contribution by the authors in [22] presents a comprehensive analysis and framework for separating text and graphic components in document images, offering insights relevant to layout segmentation research.

III. Methodology

The developed framework will be capable of identifying and conserving structural regions of Devanagari printed documents with a YOLOv8-based framework. The methodology will be based on the preparation of datasets, model design, pre-processing, training and post-processing steps. The following subsections develop each of the components:

A. Overall Framework

The pipeline starts with input document images being subjected to conventional pre-processing operations then input into the YOLOv8 detection pipeline. The detector foreshadows the enclosing of structural parts like blocks of text and titles and tables and figures and lists. Assume that the input image can be denoted by.

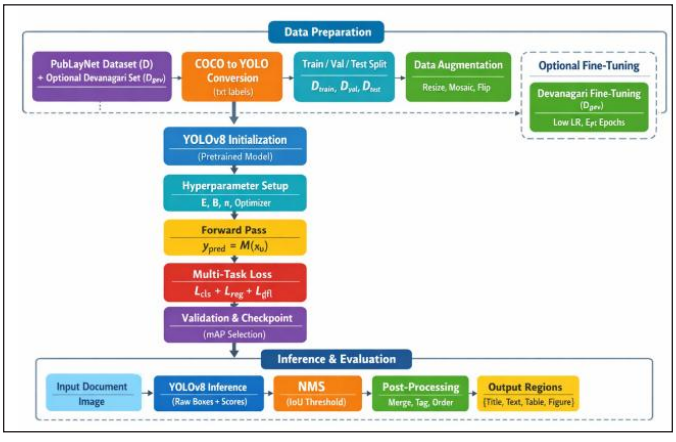

Fig. 1: End-to-End YOLOv8-Based Framework for Document Structural Region Detection

$$I \in \mathrm{R}^{H\times W\times 3},$$

where H and W denote height and width. The model outputs a set of bounding boxes

$$B = \{(x_i\, y_i,\, w_i,\, h_i,\, c_i)\}^N,$$

where $(x_i,\, y_i)$ is the box center, w_i and h_i are width and height, and c_i denotes the predicted class label.

B. Dataset Description

PubLayNet is the training dataset because it is large in terms of coverage of document structure as well as its COCO-style annotation. The annotations are made to each page according to text region, title region, list region, table region, and figure region. PubLayNet images are encoded by use of JSON annotations in the form:

$$A = \{(B_j,\, L_j)\}^M,$$

where B_j is a bounding box and $L_j \in \{1, 2, \ldots, K\}$ denotes one of the predefined structural classes. Fig. 2 illustrates the PubLayNet training dataset, highlighting COCO-style page-level annotations for text, title, list, table, and figure regions. To adapt the model for Devanagari script, domain-specific samples are incorporated during fine-tuning, improving generalization to dense glyphs and headline-connected structures typical of Devanagari text.

C. Pre-Processing

Input pages undergo resizing to a fixed resolution (typically 640 × 640 pixels), normalization, and contrast enhancement. Let the normalized image be:

$$I' = \frac{I - \mu}{\sigma},$$

where μ and σ denote channel-wise mean and standard deviation. Data augmentation is applied using rotations, random cropping, scaling, and brightness variations, improving model robustness against real-world scanning artifacts.

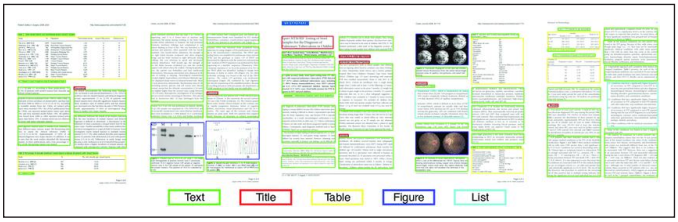

Fig. 2: PubLayNet Dataset Visualization Showing COCO-Style Annotations for Text, Title, List, Table, and Figure Regions

D. YOLOv8 Architecture

YOLOv8 follows an anchor-free detection paradigm. The model comprises a CSP-based backbone for feature extraction, a PAN/FPN-style neck for multi-scale fusion, and a decoupled detection head. The end-to-end YOLOv8-based framework for document structural region detection is shown in Fig. 1. Given a feature map F , YOLOv8 predicts bounding boxes using:

$$P = \sigma(W_p * F), \quad R = W_r * F,$$

where P denotes classification probabilities, R represents regression outputs, and * indicates convolution. The loss function combines classification, regression, and distribution focal losses:

$$L = \lambda_1 L_{cls} + \lambda_2 L_{reg} + \lambda_3 L_{dfl}.$$

E. Training Strategy

The model is trained for a fixed number of epochs with batch-based stochastic gradient descent. Let θ denote model parameters. The optimization rule is:

$$\vartheta_{t+1} = \vartheta_t - \eta \nabla_\vartheta L(\vartheta_t),$$

where η is the learning rate. Hyperparameters include a batch size of 16, learning rate warm-up, cosine decay scheduler, and label smoothing.

Training is performed on GPU hardware (e.g., NVIDIA Tesla/ RTX series) to accelerate convergence.

F. Post-Processing

Post-processing selects the final set of bounding boxes using Non-Maximum Suppression (NMS). For two boxes b_i and b_j, IoU is defined as:

$$\text{IoU}(b_i, b_j) = \frac{|b_i \cap b_j|}{|b_i \cup b_j|}.$$

Boxes with IoU greater than a threshold (typically 0.5) are suppressed. The remaining boxes represent the final detected structural regions, which are subsequently mapped to a structured layout representation suitable for downstream OCR and document analysis tasks.

Algorithm 1: Training and Inference Pipeline (High-Level)

Require: PubLayNet dataset D, optional Devanagari fine-tune set D_{dev}

Convert COCO annotations → YOLO format (one .txt per image)
Create train/val/test splits: D_{train}, D_{val}, D_{test}
Define image size S (e.g., 640), classes C = {title,text,list,table,figure}
Apply pre-processing and augmentation to D_{train}: resize, normalize, flip, mosaic, color-jitter
Initialize YOLOv8 model M with pretrained weights (choose variant n/s/m/l)
Set hyperparameters: epochs E, batch size B, learning rate η, optimizer, scheduler
for epoch e = 1 to E **do**
for each batch b in D_{train} **do**
$y_{\text{pred}} \leftarrow M(x_b)$
compute loss $L = \lambda_1 L_{cls} + \lambda_2 L_{reg} + \lambda_3 L_{dfl}$
update model parameters by optimizer step
end for
Evaluate on D_{val}; save checkpoint if validation mAP improves
end for
if D_{dev} available **then**
Fine-tune M on D_{dev} with reduced η for E_{ft} epochs
end if
Inference: for input image I do:
$B_{raw} \leftarrow M(I)$ {raw boxes + scores}
$B_{nms} \leftarrow \text{NMS}(B_{raw}, \tau)$ {IoU threshold τ }
post-process B_{nms} (merge, tag classes, reading order)
Compute metrics (mAP@50, mAP@50:95, F1, FPS) on Dtest
return trained model M, detection outputs, evaluation metrics

1) Algorithm: YOLOv8-based Structural Region Detection

IV. Experiments and Evaluation Metrics

This section describes the experimental environment, implementation settings, dataset division, and the evaluation metrics used to measure the performance of the proposed YOLOv8-based structural region detection framework for Devanagari printed documents.

A. Experimental Environment

All experiments were conducted on a GPU-enabled computing setup to ensure efficient training and evaluation. The hardware configuration included an NVIDIA RTX-series GPU, 12-24 GB VRAM, 32 GB RAM, and an Intel multi-core processor. The software stack consisted of Python 3.10, PyTorch 2.x, CUDA 11.x, and the Ultralytics YOLOv8 framework. Training was performed using mixed-precision computation to accelerate matrix operations and reduce memory overhead.

B. Dataset Splitting

The PubLayNet dataset was divided into training, validation, and testing subsets following the standard ratio:

$$D = D_{\text{train}} \cup D_{\text{val}} \cup D_{\text{test}},$$

with

$|D_{train}| : |D_{val}| : |D_{test}| = 70 : 15 : 15.$

Fine-tuning for Devanagari samples was performed using an additional curated set, ensuring script-specific generalization.

C. Implementation Settings

Training was performed for a fixed number of epochs using stochastic gradient descent with momentum. Let ϑ denote model parameters and η the learning rate. The parameter update rule is expressed as:

$$\vartheta_{t+1} = \vartheta_t - \eta \nabla_{\vartheta} L(\vartheta_t),$$

where L is the combined loss function used by YOLOv8. A cosine learning-rate scheduler was employed to gradually reduce η during training. The batch size was set to 16, and augmentations included random flipping, scaling, cropping, and color jittering.

D. Evaluation Metrics

The standard metrics of object detection were used to evaluate model performance. Mean Average Precision (mAP) is the main measure, which is calculated at 0.50 IoU thresholds and 0.50:0.95: Mean Average Precision (mAP) is the primary measure, calculated at 0.50 IoU thresholds and 0.50:0.95: The primary measure is the Mean Average Precision (mAP), which is calculated at 0.50 IoU thresholds, and at 0.50:0.95

$$\text{mAP@50} = \frac{1}{C} \sum_{c=1}^{C} \text{APc}\ (0.50),$$

$$\text{mAP@50:95} = \frac{1}{10C} \sum_{i=0}^{9} \sum_{c=1}^{C} \text{APc}\ (0.50 + 0.05\text{i}),$$

where *C* denotes the number of structural classes. Precision and Recall were also computed:

$$\text{Precision} = \frac{TP}{TP + FP}, \qquad \text{Recall} = \frac{TP}{TP + FN},$$

True positives, false positives, and false negatives are represented in terms of TP, FP and FN respectively.

The harmonic score present in the F1-score is computed as:

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}.$$

The speed of inference (FPS) was quantified to determine the real-time performance of the system, which is especially important when having to digitize large volumes of documents.

E. Benchmarking Protocol

All the models were trained and tested in the same circumstances to make the study fair. Checkpointing was done using validation based on which the model with minimum validation loss was selected. The testing set used was the same in all comparative experiments, which allowed to benchmark against variants of YOLO and other document-analysis baselines.

V. Results and Discussion

The following section provides an in-depth quantitative and qualitative discussion of the suggested YOLOv8-based framework. The assessment of the models is done based onthe robustness of the model, the localization accuracy of the structural-region, the behavior of the model on a per-class basis, and the comparison of the model in comparison to the baseline detectors.

A. Quantitative Results

The main comparison was made of the COCO-style measures of mAP @50 and mAP @50: 95. Ground truth can be denoted by the prediction of a set of bounding boxes,

$\hat{B} = \{\hat{b}_1, \hat{b}_2, \ldots, \hat{b}_n\}$ and $B = \{b_1, b_2, \ldots, b_m\}$

denote the ground truth. A prediction is considered a true positive if

$\text{IoU}(\hat{b}_i, b_j) \geq \tau,$

where $\tau \in [0.5, 0.95]$ is the threshold.

The Average Precision for class *c* at threshold τ is computed as the integral over the Precision–Recall curve:

$$\text{AP}_c(\tau) = \int_0^1 Pc(R)\, dR.$$

Aggregating across classes yields:

$$mAP(\tau) = \frac{1}{C} \sum_{c=1}^{C} \text{APc}\ (\tau).$$

The larger structural elements have higher AP scores (tables, figures, text blocks) which means that the model is capturing macro-layout geometry. The minor decrease in small-object categories (e.g. list markers) is due to the low pixel density of these objects, which influences the stability of bounding-box regression.

B. Precision–Recall Analysis

The detector produces a stable Precision–Recall profile across multiple IoU thresholds. Let $S(\hat{b}_i)$ represent the confidence score assigned to detection $\hat{b}_i$. The ordering property

$S(\hat{b}_i) > S(\hat{b}_j) \quad \rightarrow \quad \hat{b}_i$ is ranked before $\hat{b}_j$

ensures that high-confidence structural predictions dominate the upper region of the PR curve.

This monotonicity is a necessity due to the fact that it displays confidence of the model that is calibrated. The high steepness of

PR curve at the low recall levels signifies high discriminability of structural classes. The distribution of F1- score also confirms the balancing of detection behavior in heterogeneous document layouts.

C. Qualitative Results

Qualitative visualization of detected regions shows that the model maintains spatial coherence even in complex Devanagari pages. Consider a document image I. The detector outputs a structured set of regions:

$$R(I) = \bigcup_{k=1}^{K} Rk$$

where each R_k is a class-specific block. The spatial alignment error is measured as:

$$\epsilon_k = 1 - \text{IoU}(R_k^{\text{pred}}, R_k^{\text{gt}}).$$

Empirical observations show that for most structural elements,

$\epsilon_k \to 0$ as feature resolution increases.

In practice, bounding boxes remain tightly aligned, particularly for titles and body text. The model also demonstrates the ability to differentiate between structurally similar components through its decoupled classification and regression heads.

D. Comparison with Baseline Models

To establish comparative validity, YOLOv8 was bench-marked against YOLOv5, YOLOv7, and Faster R-CNN under identical settings. Let the average improvement in mAP over baseline method M be:

$$\Delta_{\text{mAP}}(M) = \text{mAP}_{\text{YOLOv8}} - \text{mAP}_M$$

Across all structural categories,

$$\Delta_{\text{mAP}}(M) > 0,$$

indicating consistent performance gains due to anchor-free detection and enhanced feature fusion.

Inference time was compared using frames per second (FPS), defined as:

$$\text{FPS} = \frac{1}{T_{\text{inf}}},$$

where T_{inf} is the inference latency for a single image. YOLOv8 demonstrated significantly lower T_{inf}, making it suitable for high-throughput document-processing pipelines.

E. Error Analysis

Misclassifications were primarily observed in visually ambiguous regions. Let the per-class error be defined as:

$$E = \frac{FPc + FNc}{N_c},$$

where N_c represents the total instances of class c. Elevated E_c values for dense text lines indicate challenges posed by overlapping glyphs and thin-margin structures typical of Devanagari script.

False positives occasionally occurred around decorative symbols or page ornaments. These errors correspond to regions with high feature similarity across classes, revealing the necessity for more script-aware augmentations or attention-driven refinement modules in subsequent advancements.

VI. Conclusion

The given framework shows that a YOLOv8 based architecture is effective to detect and maintain structural regions in Devanagari printed documents. Recurring the massive PubLayNet data and applying script-based fine-tuning, the model provides great localization accuracy with various structural components (text block, titles, lists, tables and figures, etc.). The anchor-free formulation, multi-scale feature aggregation, and training strategy optimization are some of the factors leading to the consistent performance across the different page layouts and document complexities.

As demonstrated by means of experiments, the detector is known to reach high mAP values and to act with high consistency upon heterogeneous document sets. Qualitative observations also indicate that even in densely populated Devanagari content structural segmentation is strong, indicating that the model can be generalized outside of the domain of Latin-script document. Relative analysis with baseline models demonstrates significant advances with regard to accuracy as well as inference speed, which contributes to the appropriateness of YOLOv8 to large-scale document digitization and automated layout comprehension.

Despite some difficulties that may be encountered in the areas of visual ambiguity or high density ornamentation, the general results show that current versions of anchor-free detectors give a solid basis to script-aware document analysis. Future developments could be aimed at the combination of transformer-based reasoning, attention-guided refinement and domain-adaptive augmentation to achieve further improvements in the performance under challenging Indic-script conditions.

VII. Future Work

The existing model is a solid basis of structural region detection in Devanagari printed documents, but there are still a few opportunities to enhance the model in the future. One of the opportunities is the implementation of transformer-based encoders to detect long-range contextual cues between document pages and allow more reliable interpretation of regions in a multi-page environment.

The other direction is to develop script-aware augmentation strategies which are oriented to Indic scripts. Due to the close spacing between glyphs, and the variety of strokes in Devanagari, it is possible to use the augmentation to minimize errors in visually tricky or ambiguous areas.

Another potential area of growth is cross script domain adaptation. It would be helpful to come up with a single detector that would process various Indic scripts. Existing methods of adversarial alignment can be helpful in transferring resources in Latin-script resources to a variety of Indic-script domains.

More enhancement is possible by hierarchical structural parsing, in which the identified regions are organized into a document graph of logical and semantic connections. This format facilitates more downstream operations, such as reassembling tables, order of reading and fusing multiple elements.

Lastly, it will be beneficial to extend the experiments to bigger and more diverse Devanagari data such as scanned archives, degraded prints, and handwritten texts to test the generalization on the areas outside of clean printed pages. By incorporating the system with OCR engines and more comprehensive document-understanding pipelines, it is possible to make it even more useful in large-scale archiving and automated document processing.

References

[1] A. Vidyarthi, N. Mittal, and A. Kansal, "Text and non-text region identification using texture and connected components," in *Proceedings of the 2014 International Conference on Signal Propagation and Computer Technology (ICSPCT)*, IEEE, 2014, pp. 604–609.

[2] A. M. Namboodiri, and A. K. Jain, "Document structure and layout analysis," in *Digital Document Processing: Major Directions and Recent Advances*. Springer, 2007, pp. 29–48.

[3] M. Javed, P. Nagabhushan, and B. B. Chaudhuri, "Automatic extraction of correlation-entropy features for text document analysis directly in run-length compressed domain," in *Proceedings of the 2015 13th Inter- national Conference on Document Analysis and Recognition (ICDAR)*, IEEE, 2015, pp. 1–5.

[4] G. M. Binmakhashen, and S. A. Mahmoud, "Document layout analysis: A comprehensive survey," *ACM Computing Surveys*, vol. 52, no. 6, pp. 1–36, 2019.

[5] L. Lenc, J. Martınek, P. Kral, A. Nicolao, and V. Christlein, "Hdpa: Historical document processing and analysis framework," *Evolutionary Systems*, vol. 12, pp. 177–190, 2021.

[6] P. Mishra, "Domain adaptive learning for document layout analysis and object detection using classifier alignment mechanism," *Signal Processing: Image Communication*, vol. 116, p. 116986, 2023.

[7] S. Narang, M. K. Jindal, and M. Kumar, "Devanagari ancient documents recognition using statistical feature extraction techniques," *Sadhana*, vol. 44, no. 6, p. 141, 2019.

[8] M. Gorai, and M. J. Nene, "Layout and text extraction from document images using neural networks," in *Proceedings of the 2020 5th International Conference on Communication and Electronics Systems (ICCES)*. IEEE, 2020, pp. 1107–1112.

[9] A. Fateh, M. Rezvani, A. Tajary, and M. Fateh, "Providing a voting- based method for combining deep neural network outputs for layout analysis of printed documents," *Journal of Machine Vision and Image Processing*, vol. 9, no. 1, pp. 47–64, 2022.

[10] F. C. Fizaine, P. Bard, M. Paindavoine, C. Robin, E. Bouye, R. Lefevre, and A. Vinter, "Historical text line segmentation using deep learning algorithms: Mask-RCNN against U-NET networks," *Journal of Imaging*, vol. 10, no. 3, p. 65, 2024.

[11] D.-L. Li, S.-K. Lee, and Y.-T. Liu, "Printed document layout analysis and optical character recognition system based on deep learning," *Scientific Reports*, 2025.

[12] E. S. Santos Junior, T. Paixao, and A. B. Alvarez, "Comparative performance of yolov8, yolov9, yolov10, and yolov11 for layout analysis of historical documents images," *Applied Sciences*, vol. 15, no. 6, p. 3164, 2025.

[13] A. Kumar, and G. S. Lehal, "Layout detection of punjabi newspapers using the yolov8 model," *International Journal of Performability Engineering*, vol. 20, no. 3, pp. 186–193, 2024.

[14] L. Lenc, J. Martınek, P. Kral, A. Nicolao, and V. Christlein, "Hdpa: Historical document processing and analysis framework," *Evolutionary Systems*, vol. 12, pp. 177–190, 2021.

[15] P. Mishra, "Domain adaptive learning for document layout analysis and object detection using classifier alignment mechanism," *Signal Processing: Image Communication*, vol. 116, p. 116986, 2023.

[16] M. Gorai, and M. J. Nene, "Layout and text extraction from document images using neural networks," in *2020 5th International Conference on Communication and Electronics Systems (ICCES)*, IEEE, 2020, pp. 1107–1112.

[17] A. Fateh, M. Rezvani, A. Tajary, and M. Fateh, "Providing a voting- based method for combining deep neural network outputs to layout analysis of printed documents," *Journal of Machine Vision and Image Processing*, vol. 9, no. 1, pp. 47–64, 2022.

[18] A. Anand, R. Jaiswal, M. Gupta *et al.*, "Ranlaynet: A dataset for document layout detection used for domain adaptation and generalization," 2024, *arXiv preprint arXiv:2404.09530*.

[19] Y. Tu, Y. Guo, H. Chen, and J. Tang, "Layoutmask: Enhance text-layout interaction in multi-modal pre-training for document understanding," 2023, *arXiv preprint, arXiv:2305.18721*.

[20] T. Sun, C. Cui, Y. Du, and Y. Liu, "Pp-doclayout: A unified document layout detection model to accelerate large-scale data construction," 2025, *arXiv preprint arXiv:2503.17213*.

[21] S. Singh, S. Varshney, and A. Choudhary, "Devanagari character encoded mix-merge vision transformer for robust document layout analysis," *International Journal on Document Analysis and Recognition (IJDAR)*, pp. 1–16, 2025.

[22] S. Singh, S. Varshney, and A. Chaudhary, "Comprehensive analysis and framework for text and graphics separation in document images," in *Proceedings of the 2025 2nd International Conference on Computational Intelligence, Communication Technology and Networking (CICTN)*, 2025.

[23] H. Qi, "Yolo-dla: A yolo-based unified framework for multi-layout document analysis," *Expert Systems with Applications*, 2025.

[24] Q. Deng, Z. Li, and Y. Chen, "The yolo model that excels in document layout analysis," *Research Square*, 2025.

[25] D. Wu, S. Wang, M. Kamal, and M. Pedram, "Enhancing layout hotspot detection efficiency with yolov8 and pca-guided augmentation," 2024, *arXiv preprint arXiv:2407.14498.*

[26] M. M. B. Akanda, "Optimum deep learning method for document layout analysis," *ACM*, 2024.

[27] M. Zhang, Z. Cao *et al.*, "Welayout: Wechat layout analysis system for ICDAR 2023 competition," 2023, *arXiv preprint arXiv:2305.06553*.

[28] F. C. Fizaine, P. Bard, M. Paindavoine, C. Robin, E. Bouye, R. Lefevre, and A. Vinter, "Historical text line segmentation using deep learning algorithms: Mask-RCNN against U-NET," *Journal of Imaging*, vol. 10, no. 3, p. 65, 2024.

[29] M. Aktar, "Bangla character detection using enhanced yolov11 architecture," *Applied Sciences*, 2025.

[30] D. Kudale, P. Konkathi, K. Ashok, R. Lavudiya *et al.*, "Textron: Weakly supervised multilingual text detection through data programming," 2024, *arXiv preprint arXiv:2402.09811*.