

GAZI

JOURNAL OF ENGINEERING SCIENCES

Evaluating Scene Text Recognition Models Through Multidimensional Error Analysis: Insights from Turkish Scene Text Data

Safa Dörterler^a, Emrullah Şahin^b, Durmuş Özdemir^c

Submitted: 30.10.2025 Revised: 29.11.2025 Accepted: 21.12.2025 doi:10.30855/gmbd.070525AR09

^a Kütahya Dumlupınar University, Engineering Faculty, Dept. of Computer Engineering - Kütahya, Türkiye, Orcid: 0000-0001-8778-081X^b Kütahya Dumlupınar University, Engineering Faculty, Dept. of Software Engineering - Kütahya, Türkiye, Orcid: 0000-0002-3390-6285^c Kütahya Dumlupınar University, Engineering Faculty, Dept. of Computer Engineering - Kütahya, Türkiye, Orcid: 0000-0002-9543-4076^{*}Corresponding author: safa.dorteler@dpu.edu.tr

ABSTRACT

Keywords: Optical Character recognition, scene text recognition, OCR model comparison, Turkish scene text, character substitution errors

Scene Text Recognition (STR) has emerged as a critical research area in computer vision, enabling machines to interpret textual information embedded in natural scenes. Despite significant progress with deep learning, current Optical Character Recognition (OCR) systems still face challenges in generalizing across varying languages, fonts, distortions, and environmental conditions. This study provides a comprehensive evaluation of six state-of-the-art OCR models across eight English benchmark datasets and a newly introduced Turkish scene text dataset (TS-TR). Beyond conventional metrics such as accuracy and F1-score, the analysis incorporates character-level error types and substitution patterns to reveal systematic weaknesses in model behavior. The comparative results emphasize the superiority of Transformer-based architectures, particularly MGP-STR, in diverse scene conditions, while also highlighting performance degradation in morphologically rich and non-English languages. These findings underline the need for multilingual adaptation, linguistically informed modeling, and hybrid visual language approaches to achieve robust and language-aware OCR systems applicable to real-world scenarios.

1. Introduction

Optical Character Recognition (OCR) technology has undergone decades of evolution, from early rule-based algorithms to today's deep learning models. The earliest OCR systems, dating back to the mid-20th century, relied on template matching and hand-crafted feature analysis to recognize text in scanned documents [1]. For example, the Tesseract engine developed in the 1980s relied on a sequential processing architecture, including connected-component analysis, character segmentation, and classification stages, achieving high accuracy on clean, printed text [2]. These traditional approaches, however, struggled with real-world scene text images, which often include diverse fonts, complex backgrounds, arbitrary orientations, and uneven illumination. Indeed, while mature OCR solutions work well on clean documents, most traditional methods proved ineffective for scene text recognition (STR) tasks due to the diverse text appearances and imperfect imaging conditions in real-world scenes [1].

The advent of deep learning revolutionized OCR by enabling end-to-end learning of feature representations. Modern STR models eliminate rigid stage-by-stage processing frameworks and instead adopt neural networks that directly map image pixels to character sequences in an end-to-end manner. A seminal example is the work of Shi et al. who introduced an encoder-decoder framework combining a convolutional neural network (CNN) for visual feature extraction with a

recurrent neural network (RNN) (e.g. a bi-directional LSTM) for sequence modeling and decoding [3]. This approach significantly outperformed prior hand-engineered methods, marking a turning point in the field. Subsequent improvements incorporated attention mechanisms and spatial transformer-based image rectification modules to handle perspective distortions, further boosting recognition accuracy on challenging images [1]. Over the past decade, such deep learning approaches have firmly established themselves as the dominant paradigm for STR.

In recent years, Transformer-based architectures have pushed the state-of-the-art in scene text recognition even further. Transformers, with their self-attention mechanism, have become a de facto standard for sequence modeling tasks, including text recognition [4]. Many of the latest STR models integrate Transformers either as the decoder component or throughout the encoder-decoder network, yielding superior ability to capture long-range dependencies in text sequences [4]. For instance, the TrOCR model by Li et al. employs a Vision Transformer (ViT) encoder and an autoregressive Transformer decoder, leveraging pre-trained vision and language modules to achieve highly accurate end-to-end text recognition [5]. Likewise, the recently proposed MGP-STR approach combines a ViT backbone with a novel multi-granularity prediction strategy (injecting sub-word representations from natural language processing) to implicitly incorporate linguistic context [6]. This Transformer-only STR model achieved an unprecedented ~93.3% average word accuracy on standard benchmarks, outperforming previous state-of-the-art vision-only and language-augmented methods [6]. These developments exemplify the rapid transition from CNN-RNN based architectures to Transformer-driven models that now dominate the field of OCR.

Alongside model advancements, the evaluation benchmarks for scene text have also diversified. Researchers assess OCR algorithms on a range of challenging datasets, each reflecting different real-world complexities. Early benchmarks from the ICDAR competitions (e.g. ICDAR 2003, 2013) introduced images of text in relatively controlled scenarios, whereas later contests like ICDAR 2015 focused on “incidental” scene text with more severe distortions, occlusions, and noise. Similarly, the IIIT-5K and Street View Text (SVT) datasets contain cropped word images from street signs and natural scenes, including many low-resolution and perspective-distorted examples [7]. More recent benchmarks emphasize extreme conditions: SVT-Perspective (SVTP) captures heavily perspectival text, CUTE80 was curated for curved text instances, and Total-Text features a variety of multi-oriented and curved text in diverse fonts. These datasets collectively highlight the formidable challenges in scene text recognition, such as low contrast, motion blur, low resolution, and uneven illumination, often compounded by complex backgrounds and arbitrary orientations [7,8]. A truly robust OCR model must therefore generalize across this spectrum of conditions, handling everything from well-focused frontal text to highly distorted or stylized text in the wild.

While performance metrics on standard benchmarks have steadily improved, a detailed understanding of OCR error patterns remains crucial. Character-level error analysis provides insight into systematic failure modes (e.g. confusions between visually similar characters, or consistent mistakes on certain symbols) that aggregate word-level metrics might obscure. However, the literature has largely focused on overall word accuracy, with relatively few works examining fine-grained error patterns or reporting character-level metrics. In fact, many studies propose new STR models and report impressive accuracy, but a holistic and fair comparison across methods has often been missing due to inconsistent evaluation protocols and datasets [9]. This leaves a gap in understanding why and how different algorithms fail. For instance, identifying common substitution errors (such as '0' vs 'O' or '1' vs 'l' swaps) and measuring special character error rates can pinpoint weaknesses in recognition algorithms; yet these aspects are seldom discussed in depth in prior research. The scarcity of such analyses underlines the importance of evaluating not only how often models error, but also in what ways they err – knowledge that is critical for guiding future improvements in OCR systems.

Motivated by these observations, the present work offers a comprehensive comparative study of six different OCR algorithms on a suite of eight scene-text benchmarks. The methods evaluated span from a classic rule-based engine to modern deep learning models including the open-source Tesseract OCR engine, popular deep learning frameworks like EasyOCR, Keras OCR, and DoCTR, a Transformer-based model (TrOCR), and the advanced vision Transformer model MGP-STR. By evaluating all six methods across diverse datasets (ICDAR 2003, ICDAR 2013, ICDAR 2015, IIIT5K, SVT, SVT-P, TotalText, and CUTE80) using multiple evaluation metrics (word accuracy, precision, recall, F1-score, character-level Levenshtein edit distance, special-character errors, and counts of common misrecognitions), we systematically

compare the strengths and weaknesses of each approach. The analysis highlights each algorithm's robust points and failure modes, as well as their resilience to varied text styles and image conditions. Notably, our results confirm the superior overall performance of the Transformer-based MGP-STR model across most benchmarks, while also illuminating the types of errors that even state-of-the-art models still struggle with (e.g. reading low-contrast or curved text). By providing a detailed error breakdown alongside aggregate scores, this study aims to fill the aforementioned gap in the literature offering a nuanced, character-level evaluation of modern OCR systems on challenging scene text data. Such insights are intended to guide researchers in addressing the remaining challenges of scene text processing and to pave the way for more robust and generalizable OCR solutions in the future.

This paper consists of six sections. In the second section, the historical development of OCR technology, traditional methods, deep learning-based approaches and existing literature are discussed in detail. In the third section, the OCR algorithms used, the benchmark datasets used in the experiments, the data preprocessing steps and the evaluation metrics are presented in detail. The fourth section describes the experimental setup and systematically compares the performance of different models on various datasets. The fifth section discusses the strengths and weaknesses of the existing models and analyzes the error types and performance variations based on the findings. The sixth and final section summarizes the overall conclusions of the study and makes recommendations for future research in light of the findings.

2. Literature Review

In recent years, artificial intelligence and machine learning have been widely applied across diverse domains such as medical diagnosis and biomedical risk analysis [10-12], image synthesis and generative modeling [13-15], metaheuristic and optimization-based engineering design [14,15], renewable energy applications [16], and automated classification systems in various environmental and biological contexts [17-20], demonstrating the broad impact of modern deep learning and metaheuristic approaches. Among these diverse application areas, one of the most established and extensively utilized fields is Optical Character Recognition (OCR), which focuses on the automated conversion of visual text into machine-readable form. OCR technology has a long history in converting printed documents into digital text and has achieved remarkably high accuracy rates in scanned documents [21]. In contrast, reading text from natural scene images known as Scene Text Recognition (STR) is considered significantly more challenging than traditional OCR due to factors such as complex backgrounds, variable fonts, geometric distortions, low resolution, and inconsistent illumination [22,23]. Indeed, these challenges have drawn substantial attention from both academia and industry, guiding research through competitions and benchmarks such as the ICDAR "Robust Reading" series [22]. STR can be regarded as a specialized subfield of OCR that aims to read text appearing in camera-captured images [1,23]. Early approaches to scene text recognition were typically based on character-level segmentation and hand-crafted feature extraction. These methods employed features such as Histogram of Oriented Gradients (HOG) to detect and recognize characters, which were then combined into words using language dictionaries. Wang et al. utilized HOG features and a Random Ferns classifier to detect character regions in scene images and subsequently applied a Pictorial Structures model to assemble these regions into words [24]. Similarly, several studies adopted graphical models or Conditional Random Fields (CRF) to better model dependencies between neighboring characters. In such frameworks, positional and class information of characters were jointly evaluated on a graph structure to produce more coherent word sequences [25]. However, the limited expressive power of these manually engineered features led to significant performance degradation in scenes containing complex backgrounds, low contrast, skewed alignments, or distorted text [23]. These limitations motivated the development of more flexible, data-driven deep learning approaches capable of automatically extracting discriminative features [26]. A notable milestone in this direction was the PhotoOCR system introduced by Bissacco et al. in 2013 [27]. Developed at Google, this system combined a multi-stage neural network with an integrated language model to recognize text embedded in natural scenes. By leveraging large-scale data and deep learning techniques, PhotoOCR achieved markedly higher accuracy rates compared to conventional methods of its time. Specifically, it employed a fast cascade classifier for character detection and a language model that refined predictions by exploiting contextual information, thereby improving the overall readability of scene text. The PhotoOCR framework was particularly significant in demonstrating that scene text recognition should be addressed not only as a computer vision problem but also as a linguistic and contextual understanding task [27].

Since the mid-2010s, deep learning-based approaches have rapidly become the dominant paradigm in scene text recognition. In particular, the automatic feature-learning capability of artificial neural networks has eliminated the need for manual feature engineering, leading to significant improvements in recognition accuracy [23]. One of the key innovations introduced during this period was the idea of integrating traditional rule-based components such as Hidden Markov Models (HMMs) into a single, end-to-end trainable deep network. In this context, Shi et al. proposed the Convolutional-Recurrent Neural Network (CRNN) architecture, which enabled the direct conversion of word images into corresponding character sequences in an end-to-end fashion [28]. CRNN combines a Convolutional Neural Network (CNN) for visual feature extraction with a Recurrent Neural Network (RNN) for sequential processing, allowing it to learn variable-length character sequences using the Connectionist Temporal Classification (CTC) loss function. This innovation made it possible to recognize text directly from input images without requiring explicit character segmentation, a step that had been essential in earlier approaches. The success of deep learning-based OCR systems depend not only on model architecture but also critically on the quality and diversity of the training data. In recent years, synthetic datasets capable of generating unlimited labeled samples have become widely adopted. Jaderberg et al. introduced the Synth90k dataset, containing over nine million synthetically generated word images, demonstrating that high recognition accuracy could be achieved without relying on real-world data [29]. Similarly, Gupta et al. developed the SynthText in the Wild dataset, which superimposes automatically generated text on realistic scene images, proving effective for both text detection and recognition tasks [30]. Long and Yao further extended this concept by creating the UnrealText dataset using the Unreal Engine, enabling training on highly realistic synthetic imagery [31]. Moreover, Baek et al. conducted a systematic comparative analysis of various scene text recognition architectures using SynthText datasets, examining in depth the impact of synthetic data on model performance [32]. Such synthetic datasets have since become a standard tool for improving model generalization, particularly in situations where annotated real-world data are scarce.

With the advancements in deep learning, more sophisticated architectures have emerged in the field of scene text recognition. Sequence-to-sequence (seq2seq) models based on attention mechanisms have marked a significant turning point in this domain. The Recursive Recurrent Neural Network (R2AM) model proposed by Lee and Osindero produced more coherent output sequences compared to CTC-based approaches by jointly considering both the spatial positioning of characters and their linguistic context [33]. This approach overcame the conditional independence assumption inherent in CTC, yielding more natural recognition results for irregular scene texts [23,34]. In addition, to address common issues in scene text such as perspective distortion and curved text, Shi et al. introduced a spatial transformer layer that rectifies the input image before recognition [35]. Cheng and colleagues, on the other hand, proposed the Arbitrarily-Oriented Network (AON) architecture, which processes text from multiple orientations to handle directional variability effectively [36]. Collectively, these studies initiated a new era in OCR research, where the integration of attention mechanisms, spatial transformation modules, and linguistic modeling led to substantial improvements in scene text recognition performance.

In STR systems, traditional evaluation metrics typically focus on overall accuracy rates but often fail to capture character-level errors. To address this limitation, advanced metrics such as Character-Level Evaluation (CLEval) [37] and End-to-End Evaluation Metric (EEM) [37] have been introduced. CLEval measures accuracy and deficiencies at the character level, while EEM jointly evaluates both detection and recognition components. Nevertheless, many studies continue to emphasize aggregate accuracy scores. However, character-level error analysis is crucial for identifying model weaknesses and ensuring a fairer and more comprehensive comparison between different STR systems.

Another important branch of OCR research focuses on handwriting recognition, which, similar to scene text recognition, addresses the problem of visual text interpretation; however, in this case, the input images consist of handwritten rather than printed text. This task presents additional challenges such as inter-writer variability in letter shapes, cursive writing (connected handwriting), and ambiguous character boundaries [38]. In the historical development of OCR, handwriting recognition emerged as a distinct research area, achieving significant progress through methods based on HMMs and structural feature representations [39-43]. In recent years, advances in handwriting recognition have shown a clear paradigm shift from traditional approaches toward deep learning-based architectures. This transformation is observed not only in widely used languages such as English but also in morphologically rich languages such as French. In a comprehensive review by AlKendi et al. [44], approaches for recognizing both historical and modern French handwritten

texts were analyzed in detail. The study evaluated contemporary architectures such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Convolutional Recurrent Neural Networks (CRNNs), and Transformer-based models, alongside updated variants of traditional HMM-based systems. The authors also performed comparative evaluations on benchmark datasets such as RIMES and discussed publicly available handwriting datasets and commercially used recognition systems for the French language. Barrere et al. proposed a lightweight Transformer-based architecture specifically designed for handwriting recognition, achieving performance competitive with standard LSTM-based models [45]. Likewise, attention mechanisms and encoder–decoder architectures have been increasingly employed to better model long character sequences in handwritten text recognition [44]. Recent surveys indicate that a wide range of deep learning models have been developed for handwriting recognition across different languages and scripts (e.g., Latin, Arabic, Chinese), most of which adopt CNN/LSTM or Transformer-based encoder–decoder frameworks [44,46–49]. However, several challenges remain unresolved: recognition of faded or damaged historical documents, as well as multilingual or mixed handwritten manuscripts, continues to be difficult [44]. Furthermore, the inherent variability between writers limits model generalization. Consequently, research efforts increasingly focus on data augmentation, language model integration, and writer-adaptive learning techniques, which are now regarded as critical components for advancing handwriting recognition systems [50,51].

In recent years, the field of OCR has achieved remarkable progress through deep learning and Transformer-based approaches. The transition from traditional rule-based methods to modern networks containing millions of parameters has produced groundbreaking results, particularly in addressing the challenge of reading text from natural scenes. Contemporary scene text recognition models can accurately interpret text under complex backgrounds and distortions, while handwriting recognition systems have reached near-human performance levels. Nevertheless, several open issues and research gaps persist in the literature. The lack of standardized evaluation protocols for directly comparing results across studies, the limited number of in-depth analyses on model error behavior, and the scarcity of datasets and methodologies for many languages remain as ongoing challenges. Therefore, future research should focus on character-level error analyses, fairer benchmarking practices, and the development of specialized OCR solutions tailored for low-resource languages, which will continue to be key areas of exploration for researchers.

3. Material and Methods

This section systematically presents the methodological components that constitute the experimental framework of the study. First, six different OCR algorithms evaluated in a comparative manner are introduced, and the architectural characteristics and roles of each model in scene text recognition are explained. Next, the benchmark datasets on which these algorithms were tested are described in detail, along with the preprocessing steps applied to these datasets. The subsequent subsection defines the quantitative metrics used to evaluate model performance, and finally, the details of the experimental setup are presented. The overall experimental design is summarized in Figure 1, which illustrates the datasets used, the preprocessing workflow, the evaluated OCR models, and the performance metrics applied at both the character and word levels.

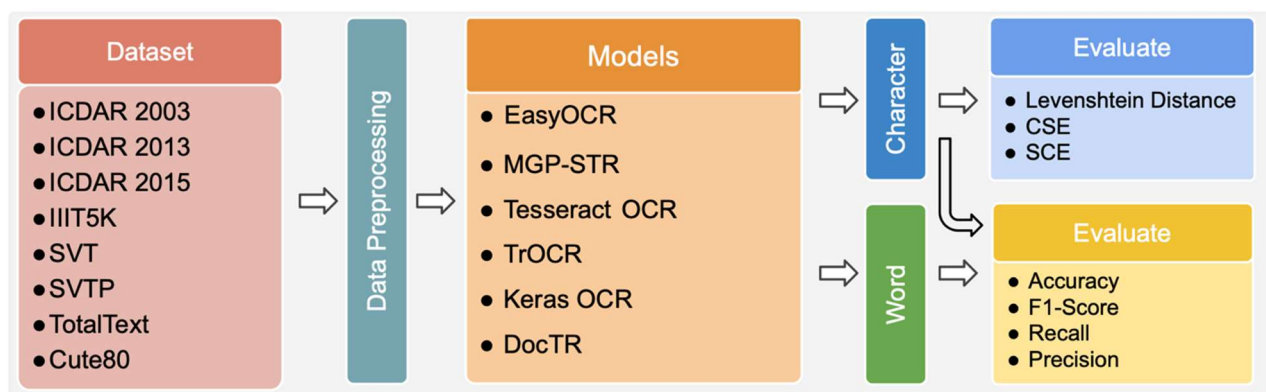


Figure 1. Graphical abstract illustrating the overall experimental workflow

3.1. OCR algorithms used

This section introduces six OCR algorithms that are widely used in scene text recognition tasks and represent distinct architectural paradigms. First, EasyOCR, which employs a two-stage CRNN-based structure, is discussed. Next, MGP-STR, a Vision Transformer-based model distinguished by its multi-granularity output strategy, is presented. The classical rule-based approach Tesseract OCR is then examined, followed by TrOCR, which provides a Transformer-based end-to-end architecture. Keras OCR, offering an easily deployable implementation of the CRNN framework, and DoCTR, designed specifically for document-oriented scene text recognition, are also evaluated. Each model is analyzed in terms of its core architectural design, its role within the OCR landscape, and its position in the related literature.

3.1.1. EasyOCR

EasyOCR is an open-source, deep learning-based framework for optical character recognition that employs a multi-stage pipeline consisting of preprocessing, text detection, recognition, and output generation [52]. In the text detection stage, it applies the CRAFT (Character Region Awareness for Text Detection) algorithm, which generates character-region and affinity maps to identify individual character areas and group them into complete text instances [53]. During the recognition stage, the detected text regions are processed using a Convolutional Recurrent Neural Network (CRNN) architecture that integrates ResNet-based convolutional feature extraction, a bidirectional Long Short-Term Memory (BiLSTM) layer for sequential dependency modeling, and Connectionist Temporal Classification (CTC) for decoding [54]. The network outputs a sequence of character probabilities, which are converted into the final text output using a “greedy” decoding strategy that selects the most probable characters in order. The recognition results can be further refined using post-processing techniques such as language models or spell correction. EasyOCR demonstrates strong performance in scene text recognition, effectively reading text under diverse fonts, orientations, and complex background conditions [44]. Moreover, its modular architecture allows easy integration of alternative text detectors and recognizers, enabling rapid adaptation to new or more advanced methods [54].

3.1.2. MGP-STR

MGP-STR (Multi-Granularity Prediction for Scene Text Recognition) is an advanced scene text recognition model that combines a Vision Transformer (ViT)-based encoder with specially designed Adaptive Addressing and Aggregation (A3) modules [55]. In this architecture, the powerful ViT backbone is complemented by three A3 modules, each designed for a different output granularity character, Byte Pair Encoding (BPE), and WordPiece allowing the model to generate text predictions simultaneously at multiple linguistic levels. For a given input image, the model produces parallel output sequences at the character, BPE, and WordPiece levels. These multi-scale outputs are then fused using a simple yet effective decision-level aggregation strategy to obtain the final recognition result. The core objective of MGP-STR is to incorporate linguistic knowledge into scene text recognition without relying on an explicit language model. To this end, the output space of the model is expanded beyond individual characters to include subword units BPE and WordPiece that are widely adopted in natural language processing, thereby enhancing the model’s ability to implicitly learn language-aware representations.

3.1.3. Tesseract OCR

Tesseract is an OCR engine originally developed by Hewlett-Packard and later released as open-source software by Google [46]. The classical Tesseract architecture detects character contours through connected-component analysis, segmenting them into components referred to as “blobs.” These blobs are then grouped at the line and word levels and partitioned according to whether the text follows fixed-width or proportional spacing [56]. The recognition process operates in two passes. In the first pass, a static classifier attempts to recognize each word, and the reliable outcomes from this stage are used to train an adaptive classifier. During the second pass, words that were not confidently recognized in the first stage are reprocessed using this newly trained adaptive classifier, leading to improved overall accuracy. Among Tesseract’s distinctive innovations are its ability to recognize inverted text (white text on a black background) through nested contour analysis, a line-finding algorithm capable of correctly identifying text lines without deskewing the page, and a quadratic

spline-based baseline alignment method that compensates for curved text lines. Remarkably, it can train a highly accurate classifier with as few as ~60,000-character samples. In modern versions (Tesseract 4 and later), the classical engine has been enhanced with a Long Short-Term Memory (LSTM)-based deep learning model, significantly improving character recognition performance [57].

3.1.4. TrOCR

The TrOCR model (Transformer-based Optical Character Recognition) is an end-to-end OCR framework that employs a Vision Transformer (ViT)-based encoder and a Transformer-based decoder architecture [5]. Unlike previous OCR approaches, TrOCR does not rely on a CNN backbone or an RNN-based language model. Instead, it operates entirely within the Transformer paradigm, generating text outputs at the wordpiece level rather than at the individual character level. The encoder component of TrOCR can be initialized with pre-trained ViT-based visual models such as DeiT or BEiT, while the decoder can be initialized with pre-trained Transformer-based language models such as RoBERTa or MiniLM [5]. Notably, TrOCR represents the first OCR framework in the literature that jointly integrates pre-trained vision and language Transformer models within a unified architecture. Through this design, the model effectively combines visual and linguistic representations learned from large-scale unlabeled data during pre-training, thereby eliminating the need for an additional external language model.

3.1.5. Keras OCR

Keras OCR is an open-source library that provides a ready-to-use, end-to-end OCR pipeline built on Keras/TensorFlow [58]. The framework integrates modern deep learning techniques for both text detection and recognition. For locating text regions, it employs a convolutional network-based detector such as CRAFT (Character Region Awareness for Text Detection) [53]. During the recognition stage, Keras OCR utilizes a Convolutional Recurrent Neural Network (CRNN)-based model to extract text sequences directly from images [28]. This structure enables highly accurate text extraction from both scene and document images, and the model can be fine-tuned with custom datasets for task-specific adaptation. Owing to its ease of use and flexibility, Keras OCR is widely preferred for prototyping, benchmarking, and comparative evaluation in OCR research.

3.1.6. DocTR

DoCTR (Document Text Recognition) is a document-oriented, deep learning-based OCR library that employs a two-stage end-to-end architecture [58]. In the first stage, it detects text regions across the entire page; in the second, it recognizes the textual content within each detected region. The text detection process adopts a segmentation-based approach, utilizing a Convolutional Neural Network (CNN)-based text detector with the Differentiable Binarization (DB) technique to capture text of arbitrary orientations [59]. During the recognition stage, the cropped text regions are sequentially processed and converted into character sequences using a CRNN or a similar attention-based network. DoCTR focuses particularly on achieving high-accuracy text reading in scanned documents while preserving page layout integrity. By integrating multiple advanced OCR models under a unified framework, DoCTR provides a versatile and efficient foundation for comprehensive document analysis research.

3.2. Dataset

The primary benchmark datasets commonly used in scene text recognition research are summarized below. In this study, eight major scene text datasets were utilized: ICDAR 2003 [60, 61], ICDAR 2013 [62, x"63], ICDAR 2015 [64], IIIT5K [64], Street View Text (SVT) [64], Street View Text Perspective (SVTP) [64], TotalText [65], and CUTE80 [66]. These datasets are widely regarded as standard benchmarks for evaluating Scene Text Recognition (STR) systems under varying difficulty conditions. Table 1 presents a comparative overview of these eight benchmark datasets. It outlines the key characteristics of each dataset, including the total number of images, predominant text orientation, intended task (detection, recognition, or end-to-end OCR), language of the text, and the major recognition challenges associated with each.

Table 1. A comparative summary of widely used datasets in OCR research

Dataset Name	Total Images	Text Type	Intended Task	Language	Challenges
ICDAR 2003	507	Horizontal	Detection, Recognition	English	Background clutter, lighting variations, diverse fonts
ICDAR 2013	462	Horizontal	Detection, Recognition, End-to-End	English	Complex backgrounds, varied fonts and sizes
ICDAR 2015	1500	Multi-oriented (perspective)	Detection, Recognition, End-to-End	English	Perspective distortion, blur, low resolution
IIIT5K	5000	Horizontal	Recognition	English	Diverse fonts and styles, some low-res samples
SVT	904	Horizontal	Detection, Recognition, End-to-End	English	Blur, noise, low resolution
SVTP	645	Perspective (angled)	Recognition	English	Perspective distortion, slanted text
TotalText	1555	Curved & multi-oriented	Detection, Recognition, End-to-End	English	Curved text, multi-directional layout, background clutter
CUTE80	288	Curved	Recognition	English	Curved text (character deformation), background clutter

3.3. Preprocessing

The test data from the eight scene text datasets used in this study (CUTE80, ICDAR2003, ICDAR2013, ICDAR2015, IIIT5K, SVT, SVTP, and TotalText) were subjected to a standardized preprocessing procedure. Since no training was performed, only the test images were processed. Using the coordinates provided in the annotation files, text regions were cropped, and invalid or unreadable labels were removed. The cleaned text samples were then matched with their corresponding cropped sub-images and saved in CSV format. These operations were applied consistently across all datasets, thereby establishing a unified testing framework for evaluating the OCR models.

3.4. Evaluation metrics

The models used in this study were evaluated using standard performance metrics. The following equations define these metrics used to assess model effectiveness [67].

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+F} \quad (1)$$

$$\text{Precision} = \frac{TP}{TP+FP} \quad (2)$$

$$\text{Recall} = \frac{TP}{TP+FN} \quad (3)$$

$$\text{F1 Score} = \frac{2\text{PrecisionRecall}}{\text{Precision} + \text{Recall}} \quad (4)$$

In the context of performance evaluation, True Positive (TP) refers to cases where the model correctly predicts a positive instance, while True Negative (TN) denotes correctly identified negative cases. False Positive (FP) describes instances that are incorrectly classified as positive despite being negative, and False Negative (FN) corresponds to cases where positive instances are wrongly predicted as negative.

In addition to conventional evaluation metrics, several advanced error metrics were employed to enable a more detailed analysis of the error types produced by the OCR systems.

Levenshtein Distance

The Levenshtein distance represents the minimum number of edit operations insertions, deletions, or substitutions required to transform one character sequence into another. In this study, it directly reflects the total number of character-

level errors between the model outputs and the corresponding ground-truth labels.

$$D_{i,j} = \min \begin{cases} D_{i-1,j} + 1 & \text{(deletion)} \\ D_{i,j-1} + 1 & \text{(insertion)} \\ D_{i-1,j-1} + \text{cost} & \text{(substitution)} \end{cases} \quad (5)$$

Here, $D_{i,j}$ denotes the distance between the i -th character of the first sequence and the j -th character of the second sequence. If the compared characters are identical, the cost is set to 0; otherwise, the cost is set to 1.

Special Character Error Rate

$$\text{SCER} = \frac{\text{SpecialCharacterErrors}}{\text{TotalSpecialCharacters}} \quad (6)$$

Common Substitution Errors

This qualitative metric identifies the character-level misclassifications frequently made by the OCR system. Common substitution patterns such as confusing the letter “O” with the digit “0” or the character “l” with “1” were analyzed to reveal typical recognition errors.

4. Results

This section presents the performance results of the six evaluated OCR algorithms across eight scene text datasets. The analyses were conducted along two main dimensions character-level and word-level with each metric designed to provide insights not only into overall system accuracy but also into specific error patterns. First, general model performances were compared using character-based metrics such as accuracy, F1-score, and Levenshtein distance. Subsequently, word-level evaluations were performed using comparable measures. Finally, systematic error types including special character errors and common character substitutions were analyzed in detail to reveal the strengths and weaknesses of each model. The findings enable a quantitative assessment of how robust each system is when confronted with diverse text structures and visual conditions.

All experiments were conducted on a system equipped with an Apple M2 Pro processor and 16 GB unified memory. The evaluation workflow was implemented using Python 3.10, and all OCR models were executed with their official open-source implementations. Preprocessing, model inference, and metric calculations were performed using standard Python toolkits.

4.1. Character-level results

The performance of optical character recognition systems can be more accurately assessed not only through word-level evaluations but also through detailed character-level analyses. In this section, each model’s character recognition performance is examined using fundamental metrics such as Accuracy, Precision, Recall, and F1-Score, as well as the Levenshtein Distance, which measures discrepancies between predicted and ground-truth character sequences. Additionally, each model’s overall recognition capability is supported by Total and Correct character counts, while numerical differences are visually compared using graphical representations. The character-level performance results for the ICDAR 2003 dataset are presented in Table 2.

The MGP-STR model achieved the highest performance across all metrics in the character-level evaluations conducted on the ICDAR 2003 dataset. With an accuracy of 97.06% and an F1-score of 95.62%, the model outperformed all others by a significant margin. In addition to its superior accuracy, it also produced the lowest Levenshtein distance (165), indicating minimal character-sequence errors. Keras OCR ranked second with an accuracy of 88.11%, while EasyOCR demonstrated moderate performance with 79.63% accuracy. In contrast, Tesseract OCR yielded the lowest scores across all metrics, proving inadequate for reliable character recognition tasks. These findings suggest that MGP-STR possesses a

robust and consistent prediction capability, particularly on datasets containing scene texts with low variance.

Table 2. Character-level OCR model performance – ICDAR 2003

Metric	EasyOCR	MGP-STR	Tesseract OCR	TrOCR	Keras OCR	DoCTR
Accuracy (%)	0.7963	0.9706	0.3283	0.6412	0.8811	0.7686
F1 Score (%)	0.7435	0.9562	0.2765	0.5832	0.8457	0.6735
Recall (%)	0.7400	0.9523	0.2751	0.5909	0.8386	0.6668
Precision (%)	0.7471	0.9602	0.2779	0.5757	0.8529	0.6803
Levenshtein Distance	1120	165	3656	2542	679	1286
Total / Correct	5415/4312	5415/5256	5415/1778	5415/3472	5415/4771	5415/4162

The character-level performance results for the ICDAR 2013 dataset are presented in Table 3.

Table 3. Character-level OCR model performance – ICDAR 2013

Metric	EasyOCR	MGP-STR	Tesseract OCR	TrOCR	Keras OCR	DoCTR
Accuracy (%)	0.7899	0.9692	0.2288	0.6885	0.8748	0.7634
F1 Score (%)	0.7777	0.9782	0.1951	0.6569	0.8754	0.7299
Recall (%)	0.7703	0.9669	0.1926	0.6638	0.8624	0.7162
Precision (%)	0.7853	0.9898	0.1976	0.6502	0.8888	0.7441
Levenshtein Distance	981	143	3563	2091	598	1108
Total / Correct	4616/3646	4616/4474	4616/1056	4616/3178	4616/4038	4616/3524

The MGP-STR model achieved the highest performance across all metrics in the character-level evaluations conducted on the ICDAR 2013 dataset. With an accuracy of 96.92% and an F1-score of 97.82%, it delivered the best results among all evaluated models, while also yielding the lowest Levenshtein distance (143), indicating minimal sequence-editing errors. Keras OCR ranked second with 87.48% accuracy, and EasyOCR showed moderate performance with 78.99% accuracy. In contrast, Tesseract OCR demonstrated the weakest results, with only 22.88% accuracy and an F1-score of 19.51%, highlighting its limitations in character recognition tasks. These findings reaffirm that MGP-STR possesses a strong generalization capability across diverse scene types.

The character-level performance results for the ICDAR 2015 dataset are presented in Table 4.

Table 4. Character-level OCR model performance – ICDAR 2015

Metric	EasyOCR	MGP-STR	Tesseract OCR	TrOCR	Keras OCR	DoCTR
Accuracy (%)	0.4178	0.8991	0.2609	0.3408	0.6623	0.6544
F1 Score (%)	0.4198	0.9098	0.2449	0.3207	0.6793	0.6555
Recall (%)	0.4144	0.8965	0.2405	0.3269	0.6567	0.6413
Precision (%)	0.4254	0.9235	0.2495	0.3146	0.7036	0.6704
Levenshtein Distance	6476	1140	8287	9432	3816	3905
Total / Correct	11109/4641	11109/9988	11109/2898	11109/3786	11109/7357	11109/7270

As shown in Table 4, the MGP-STR model demonstrated clear superiority across all metrics in the character-level evaluations on the ICDAR 2015 dataset. With an accuracy of 89.91% and an F1-score of 90.98%, the model achieved both high accuracy and consistency by maintaining the lowest Levenshtein distance (1140) among all evaluated methods. Keras OCR ranked second with 66.23% accuracy and an F1-score of 67.93%, while EasyOCR and TrOCR showed comparatively lower performance on this dataset. In particular, Tesseract OCR produced weak results, achieving only 26.09% accuracy with a high Levenshtein distance (8287). These results indicate that MGP-STR is capable of maintaining robust performance even on degraded and challenging scene text datasets.

The character-level performance results for the IIIT5K dataset are presented in Table 5.

Table 5. Character-level OCR model performance – IIIT5K

Metric	EasyOCR	MGP-STR	Tesseract OCR	TrOCR	Keras OCR	DoCTR
Accuracy (%)	0.7956	0.9800	0.4708	0.6286	0.8955	0.8317
F1 Score (%)	0.7717	0.9858	0.4116	0.5863	0.8895	0.7893
Recall (%)	0.7738	0.9842	0.4149	0.6020	0.8877	0.7888
Precision (%)	0.7697	0.9874	0.4083	0.5714	0.8914	0.7899
Levenshtein Distance	3233	303	8246	6995	1790	2820
Total / Correct	15253/12135	15253/14696	15253/7181	15253/9426	15253/13657	15253/12686

As shown in Table 5, the MGP-STR model achieved by far the highest character-level performance on the IIIT5K dataset. With an accuracy of 98.00% and an F1-score of 98.58%, the model delivered nearly error-free recognition, while also recording the lowest Levenshtein distance (303), corresponding to only a minimal number of edit operations. These results demonstrate the model's exceptional effectiveness in character-level recognition, even under complex and variable scene conditions. Keras OCR achieved consistent results with 89.55% accuracy, whereas EasyOCR and TrOCR performed at relatively lower levels. Tesseract OCR, similar to its performance on other datasets, again showed weak results with high error rates. The character-level performance results for the Street View Text (SVT) dataset are presented in Table 6.

Table 6. Character-level OCR model performance – SVT

Metric	EasyOCR	MGP-STR	Tesseract OCR	TrOCR	Keras OCR	DoCTR
Accuracy (%)	0.6499	0.9795	0.3930	0.4642	0.8625	0.7742
F1 Score (%)	0.6477	0.9814	0.3686	0.4353	0.8615	0.7689
Recall (%)	0.6423	0.9779	0.3673	0.4476	0.8543	0.7559
Precision (%)	0.6532	0.9850	0.3700	0.4236	0.8689	0.7823
Levenshtein Distance	1329	78	2354	3154	523	860
Total / Correct	3796/2467	3796/3718	3796/1492	3796/1762	3796/3274	3796/2939

In the character-level analyses conducted on the Street View Text (SVT) dataset, the MGP-STR model once again demonstrated outstanding performance, surpassing all other models across every metric. With an accuracy of 97.95% and an F1-score of 98.14%, the model exhibited high consistency in scene text recognition, while achieving the lowest Levenshtein distance (78), indicating minimal correction requirements. Keras OCR ranked second with 86.25% accuracy and an F1-score of 86.15%, whereas EasyOCR delivered moderate performance. TrOCR and Tesseract OCR, on the other hand, showed weaker results, characterized by low accuracy and high error rates. These findings highlight the strong generalization capability of MGP-STR, even in natural scene environments such as street signs.

The character-level performance results for the Street View Text Perspective (SVTP) dataset are presented in Table 7.

Table 7. Character-level OCR model performance – SVTP

Metric	EasyOCR	MGP-STR	Tesseract OCR	TrOCR	Keras OCR	DoCTR
Accuracy (%)	0.4515	0.9283	0.1979	0.3260	0.6965	0.6440
F1 Score (%)	0.4615	0.9508	0.1886	0.3210	0.7226	0.6684
Recall (%)	0.4486	0.9395	0.1844	0.3254	0.6995	0.6496
Precision (%)	0.4752	0.9623	0.1931	0.3166	0.7472	0.6884
Levenshtein Distance	2120	277	3107	3411	1184	1387
Total / Correct	3865/1745	3865/3588	3865/765	3865/1260	3865/2692	3865/2489

In the SVTP (Street View Text Perspective) dataset characterized by challenging perspective distortions and orientation variations the MGP-STR model once again exhibited outstanding character recognition performance, outperforming all other models by a wide margin. With an accuracy of 92.83% and an F1-score of 95.08%, MGP-STR achieved the most precise results, accompanied by the lowest Levenshtein distance (277), indicating near error-free character alignment. Keras OCR also demonstrated consistent performance on this dataset, achieving 69.65% accuracy and an F1-score of 72.26%. In contrast, EasyOCR and TrOCR achieved only moderate results, while Tesseract OCR performed poorly due to its inability to effectively handle the skewed and distorted text samples characteristic of SVTP. These findings confirm that MGP-STR not only excels in regular scene text recognition but also maintains strong generalization capability under severe geometric variations.

The character-level performance results for the TotalText dataset are presented in Table 8.

Table 8. Character-level OCR model performance – TotalText

Metric	EasyOCR	MGP-STR	Tesseract OCR	TrOCR	Keras OCR	DoCTR
Accuracy (%)	0.5088	0.8601	0.2313	0.3953	0.6882	0.5427
F1 Score (%)	0.5061	0.8847	0.2104	0.3766	0.6998	0.5352
Recall (%)	0.5016	0.8698	0.2070	0.3794	0.6888	0.5189
Precision (%)	0.5107	0.9000	0.2139	0.3738	0.7113	0.5526
Levenshtein Distance	6049	1730	9416	8700	4027	5769
Total / Correct	12108/6161	12108/10414	12108/2800	12108/4786	12107/8332	12108/6571

The TotalText dataset poses unique challenges for scene text recognition due to its inclusion of inclined and curved text structures. Despite this complexity, the MGP-STR model achieved the highest character recognition performance, with an accuracy of 86.01% and an F1-score of 88.47%. Moreover, it attained the lowest Levenshtein distance (1730), indicating minimal edit errors even under complex text configurations. Keras OCR achieved a reasonable performance with 68.82% accuracy, though its character-level error rate remained significantly higher compared to MGP-STR. EasyOCR and DoCTR delivered moderate performance results, while Tesseract OCR and TrOCR showed weak recognition capabilities with low accuracy and high error rates, struggling to decode the multi-oriented text structures present in TotalText. These findings demonstrate that MGP-STR exhibits strong generalization capacity at the character level, even when dealing with highly variable and complex scene texts.

The character-level performance results for the CUTE80 dataset are presented in Table 9.

Table 9. Character-level OCR model performance – CUTE80

Metric	EasyOCR	MGP-STR	Tesseract OCR	TrOCR	Keras OCR	DoCTR
Accuracy (%)	0.5883	0.8671	0.2445	0.4076	0.7750	0.5351
F1 Score (%)	0.6359	0.9327	0.2419	0.4263	0.7796	0.5295
Recall (%)	0.6250	0.9104	0.2366	0.4250	0.7642	0.5091
Precision (%)	0.6471	0.9561	0.2474	0.4276	0.7956	0.5515
Levenshtein Distance	709	225	1305	1092	418	829
Total / Correct	1693/996	1693/1468	1693/414	1693/690	1693/1312	1693/906

The CUTE80 dataset, known for its curved text samples, provides a challenging test environment for scene text recognition tasks. Under these conditions, the MGP-STR model achieved the highest performance across all metrics, with an accuracy of 86.71% and an F1-score of 93.27%. It also achieved the lowest Levenshtein distance (225), marking it as the most accurate model in terms of minimal edit errors. Keras OCR demonstrated consistent second-place performance with 77.50% accuracy and an F1-score of 77.96%, while EasyOCR and DoCTR produced comparatively lower results. TrOCR, and especially Tesseract OCR, performed the weakest, yielding low accuracy and high error rates when confronted with the curved and complex text structures characteristic of this dataset.

These findings confirm that MGP-STR possesses a strong generalization capability not only for traditional and modern scene texts but also for curved and irregular text patterns.

The comparative character-level accuracy performance of all OCR models across different datasets is illustrated using radar charts (Figure 2).

In the comparative analysis presented in Figure 2, the MGP-STR model achieved the highest accuracy rates across all datasets, while Keras OCR consistently ranked second overall. EasyOCR and DoCTR demonstrated moderate performance levels, whereas Tesseract OCR exhibited notably low accuracy across all datasets. In relatively more legible datasets such as IIIT5K and SVT, all models attained higher accuracy values; however, in more challenging datasets like SVTP and TotalText, a significant decline in overall accuracy rates was observed.

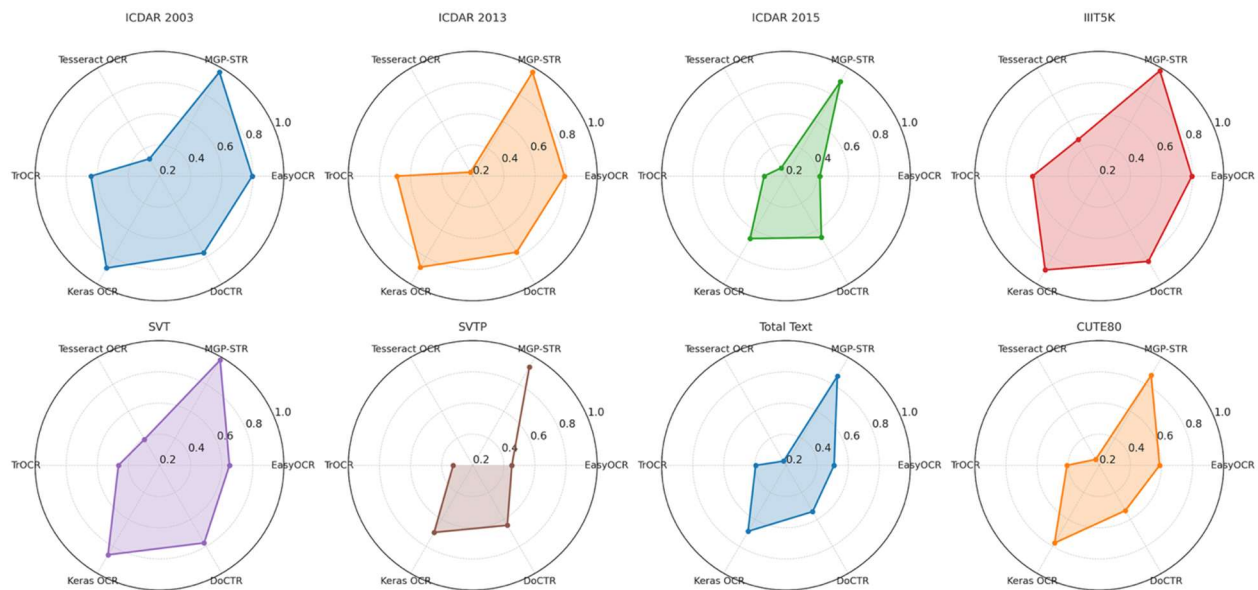


Figure 2. Radar chart comparison of character-level accuracy performance of OCR models across eight different scene text datasets

4.2. Word-Level Results

The practical effectiveness of OCR systems is most evaluated in terms of word-level accuracy, which directly reflects end-user performance. Therefore, in this section, word-level accuracy analyses were conducted for each of the evaluated models. The ratios were computed based on the number of correctly recognized words relative to the total number of words processed. These analyses are crucial for understanding how well character-level performance translates into word-level recognition accuracy.

The word-level error metrics and performance results for the ICDAR 2003 dataset are presented in Table 10.

Table 10. Word-level OCR model performance – ICDAR 2003

Metric	EasyOCR	MGP-STR	Tesseract OCR	TrOCR	Keras OCR	DoCTR
Accuracy (%)	0.6477	0.9088	0.2421	0.4408	0.7218	0.5682
F1 Score (%)	0.6477	0.9088	0.2421	0.4408	0.7218	0.5682
Recall (%)	0.6477	0.9088	0.2421	0.4408	0.7218	0.5682
Precision (%)	0.6477	0.9088	0.2421	0.4408	0.7218	0.5682
Total / Correct	1107/717	1107/1006	1107/268	1107/488	1107/799	1107/629

In the ICDAR 2003 dataset, the MGP-STR model achieved the best performance by correctly recognizing 1006 out of 1107 words, corresponding to a word-level accuracy of 90.88%. This high success rate aligns with the character-level results, demonstrating the model's strong overall recognition capability. Keras OCR ranked second with an accuracy of 72.18%, while Tesseract OCR achieved only 268 correct recognitions, resulting in a notably low accuracy of 24.21%.

The word-level error metrics and performance results for the ICDAR 2013 dataset are presented in Table 11.

Table 11. Word-level OCR model performance – ICDAR 2013

Metric	EasyOCR	MGP-STR	Tesseract OCR	TrOCR	Keras OCR	DoCTR
Accuracy (%)	0.6380	0.8785	0.1427	0.4941	0.6711	0.5390
F1 Score (%)	0.6380	0.8785	0.1427	0.4941	0.6711	0.5390
Recall (%)	0.6380	0.8785	0.1427	0.4941	0.6711	0.5390
Precision (%)	0.6380	0.8785	0.1427	0.4941	0.6711	0.5390
Total / Correct	757/483	757/665	757/108	757/374	757/508	757/408

In the ICDAR 2013 dataset, word-level analyses revealed that the MGP-STR model achieved the highest performance, with a word accuracy of 87.85%. Out of 757 test words, the model correctly predicted 665, maintaining its superiority

from character-level recognition at the word level as well. Keras OCR ranked second with 67.11% accuracy, while EasyOCR demonstrated reasonable performance with 63.80% accuracy. In contrast, Tesseract OCR produced the weakest results, correctly recognizing only 108 words with an overall accuracy of 14.27%. Examination of total and correctly predicted word counts clearly highlights the consistency and robustness of the MGP-STR model.

The word-level error metrics and performance results for the ICDAR 2015 dataset are presented in Table 12.

Table 12. Word-level OCR model performance – ICDAR 2015

Metric	EasyOCR	MGP-STR	Tesseract OCR	TrOCR	Keras OCR	DoCTR
Accuracy (%)	0.2700	0.7786	0.1574	0.1479	0.4127	0.4642
F1 Score (%)	0.2724	0.7856	0.1589	0.1493	0.4165	0.4683
Recall (%)	0.2723	0.7855	0.1589	0.1493	0.4165	0.4682
Precision (%)	0.2725	0.7857	0.1589	0.1493	0.4165	0.4685
Total / Correct	2096/566	2096/1632	2096/330	2096/310	2096/865	2096/973

In the ICDAR 2015 dataset, the MGP-STR model achieved by far the highest word recognition performance, with an accuracy of 77.86% and 1632 correctly recognized words. This outcome demonstrates that the model's strong character-level performance is consistently reflected at the word level. Keras OCR and DoCTR achieved moderate results, with 41.27% and 46.42% accuracy, respectively, while EasyOCR performed lower with 27.00% accuracy. Tesseract OCR and TrOCR were the weakest models in this dataset, correctly recognizing only 330 and 310 words, respectively. These findings once again confirm the robust generalization capability of MGP-STR, particularly in datasets containing complex and distorted scene text. The word-level error metrics and performance results for the IIIT5K dataset are presented in Table 13.

Table 13. Word-level OCR model performance – IIIT5K

Metric	EasyOCR	MGP-STR	Tesseract OCR	TrOCR	Keras OCR	DoCTR
Accuracy (%)	0.6313	0.9577	0.3143	0.3659	0.7488	0.6333
F1 Score (%)	0.6313	0.9577	0.3143	0.3659	0.7488	0.6333
Recall (%)	0.6313	0.9577	0.3143	0.3659	0.7488	0.6333
Precision (%)	0.6313	0.9577	0.3143	0.3659	0.7488	0.6333
Total / Correct	3000/1894	2952/2827	3000/943	2952/1080	2998/2245	3000/1900

In the IIIT5K dataset, word-level analyses revealed that the MGP-STR model achieved the highest performance, with an accuracy of 95.77% and 2827 correctly recognized words. This result demonstrates that the model produces highly reliable outputs not only at the character level but also at the word level. Keras OCR ranked second with 74.88% accuracy, while EasyOCR and DoCTR exhibited comparable performance, achieving 63.13% and 63.33% accuracy, respectively. Tesseract OCR and TrOCR yielded relatively lower success rates, correctly recognizing 943 and 1080 words, respectively. These findings further confirm the superiority of MGP-STR in terms of both accuracy and consistency when recognizing scene text.

The word-level error metrics and performance results for the SVT dataset are presented in Table 14.

Table 14. Word-level OCR model performance – SVT

Metric	EasyOCR	MGP-STR	Tesseract OCR	TrOCR	Keras OCR	DoCTR
Accuracy (%)	0.5100	0.9474	0.2968	0.2334	0.6862	0.6105
F1 Score (%)	0.5100	0.9474	0.2968	0.2334	0.6862	0.6105
Recall (%)	0.5100	0.9474	0.2968	0.2334	0.6862	0.6105
Precision (%)	0.5100	0.9474	0.2968	0.2334	0.6862	0.6105
Total / Correct	647/330	647/613	647/192	647/151	647/444	647/395

Word-level analyses conducted on the SVT dataset revealed that the MGP-STR model achieved the highest accuracy, not only in character recognition but also in maintaining full word integrity. With an accuracy of 94.74% and 613 correctly recognized words, the model demonstrated strong resilience to the natural variations found in scene text. Keras OCR followed with a relatively strong performance of 68.62% accuracy, while EasyOCR and DoCTR achieved moderate results

with 51.00% and 61.05% accuracy, respectively. Tesseract OCR and TrOCR, however, showed below-expected performance, characterized by low accuracy and a limited number of correctly recognized words. This dataset, which includes curved, small, and low-contrast text against complex backgrounds, clearly illustrates how well-performing models like MGP-STR can adapt to the diverse conditions of real-world scene text.

The word-level error metrics and performance results for the SVTP dataset are presented in Table 15.

Table 15. Word-level OCR model performance – SVTP

Metric	EasyOCR	MGP-STR	Tesseract OCR	TrOCR	Keras OCR	DoCTR
Accuracy (%)	0.2910	0.8449	0.1137	0.1625	0.4328	0.4431
F1 Score (%)	0.3045	0.8842	0.1190	0.1705	0.4543	0.4645
Recall (%)	0.3036	0.8817	0.1186	0.1705	0.4543	0.4639
Precision (%)	0.3054	0.8868	0.1194	0.1705	0.4543	0.4651
Total / Correct	677/197	677/572	677/77	677/110	677/293	677/300

The SVTP dataset, which primarily consists of scene texts exhibiting significant perspective distortions, serves as a crucial benchmark for evaluating OCR systems. Under these challenging conditions, the MGP-STR model achieved the highest performance by a clear margin, with an accuracy of 84.49% and 572 correctly recognized words. This result demonstrates that the model's consistent character-level accuracy extends effectively to the word level, further confirming its robustness against visual variations. Keras OCR and DoCTR achieved moderate performances with 43.28% and 44.31% accuracy, respectively. In contrast, more traditional or hybrid models such as EasyOCR, Tesseract OCR, and TrOCR performed poorly on this dataset. These findings once again confirm the clear superiority of modern deep learning-based architectures in handling highly deformed and angled text scenarios.

The word-level error metrics and performance results for the TotalText dataset are presented in Table 16.

Table 16. Word-level OCR model performance – TotalText

Metric	EasyOCR	MGP-STR	Tesseract OCR	TrOCR	Keras OCR	DoCTR
Accuracy (%)	0.3639	0.7785	0.1284	0.2021	0.4921	0.3531
F1 Score (%)	0.3639	0.7785	0.1284	0.2021	0.4921	0.3531
Recall (%)	0.3639	0.7785	0.1284	0.2021	0.4921	0.3531
Precision (%)	0.3639	0.7785	0.1284	0.2021	0.4921	0.3531
Total / Correct	2212/805	2212/1722	2212/284	2212/447	2211/1088	2212/781

The TotalText dataset presents a highly challenging benchmark for scene text recognition due to its multi-oriented text alignments, curved word structures, and complex backgrounds. Despite these difficulties, the MGP-STR model emerged as the top-performing system, achieving an accuracy of 77.85% with 1722 correctly recognized words. This outstanding result demonstrates the model's ability to generalize effectively not only on straight text but also on curved and multi-directional scene layouts. Keras OCR delivered a moderate performance with 49.21% accuracy, while EasyOCR and DoCTR achieved more limited success with 36.39% and 35.31% accuracy, respectively. Tesseract OCR and TrOCR struggled to adapt to the dataset's complex spatial configurations, producing notably low accuracy scores.

The word-level error metrics and performance results for the CUTE80 dataset are presented in Table 17.

Table 17. Word-level OCR model performance – CUTE80

Metric	EasyOCR	MGP-STR	Tesseract OCR	TrOCR	Keras OCR	DoCTR
Accuracy (%)	0.4548	0.7968	0.1161	0.1935	0.4677	0.2645
F1 Score (%)	0.4870	0.8495	0.1250	0.2075	0.5026	0.2847
Recall (%)	0.4844	0.8414	0.1250	0.2066	0.5017	0.2847
Precision (%)	0.4896	0.8576	0.1250	0.2083	0.5035	0.2847
Total / Correct	310/141	310/247	310/36	310/60	310/145	310/82

The CUTE80 dataset is a specialized benchmark characterized by curved and irregularly shaped scene texts. Despite its

structural complexity, the MGP-STR model achieved the highest performance, with an accuracy of 79.68% and 247 correctly recognized words. The model's ability to effectively decode such curved text structures highlights the flexibility and contextual processing capacity of its Transformer-based architecture. Keras OCR ranked second with 46.77% accuracy, while EasyOCR showed comparable performance with 45.48% accuracy. Models such as DoCTR, TrOCR, and Tesseract OCR struggled to recognize the slanted and stylized texts in this dataset, yielding relatively low accuracy rates. These findings emphasize that a model's performance is influenced not only by textual content but also by the structural form of the text itself.

The word-level accuracy performances of all OCR models across eight scene text datasets are comparatively illustrated using radar charts in Figure 3.

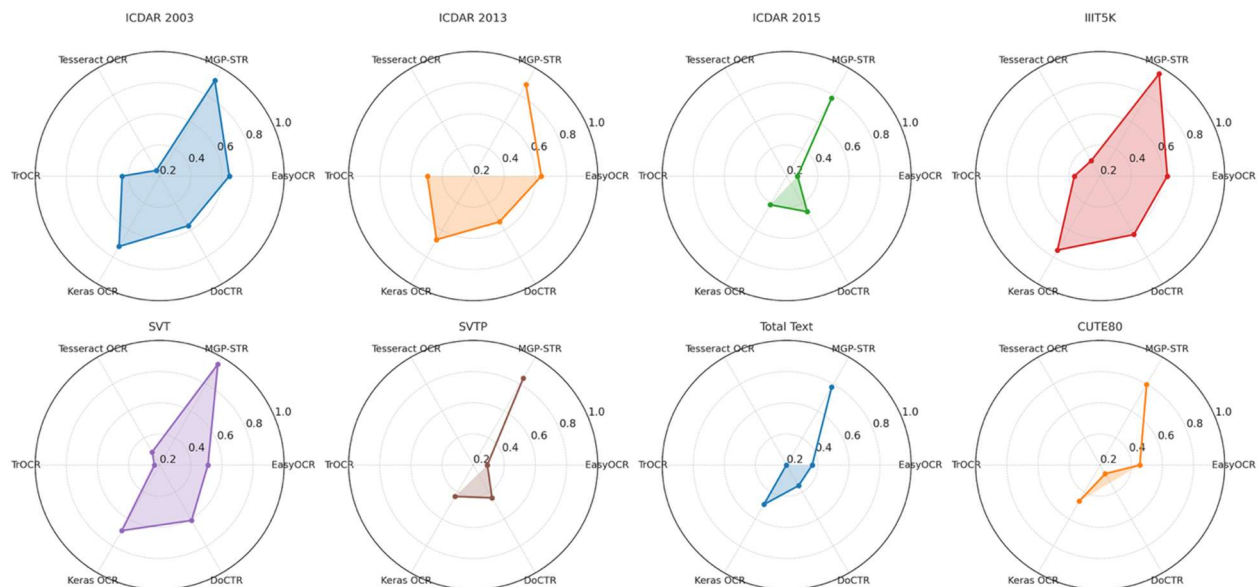


Figure 3. Radar chart comparison of word-level accuracy performance of OCR models across eight different scene text datasets

As shown in Figure 3, a pattern similar to the character-level results is also observed at the word level. The MGP-STR model consistently achieved the highest word accuracy across all datasets, maintaining its leading position. Keras OCR and EasyOCR produced competitive results in most datasets, whereas the word-level performance of Tesseract OCR remained considerably lower. In challenging scene text datasets such as SVTP, ICDAR 2015, and TotalText a noticeable decline in accuracy was observed for all models. However, MGP-STR and Keras OCR demonstrated relatively higher resilience under these difficult conditions. These results further reinforce the clear superiority of modern deep learning-based OCR models over traditional recognition systems.

4.3. Error Type Analysis

While accuracy rates serve as a fundamental metric in the evaluation of optical character recognition systems, analyzing the types of errors produced by the models provides a deeper understanding of their overall performance. In particular, within Scene Text Recognition (STR) tasks, the systematic confusion of visually similar characters exposes inherent architectural limitations of the models. In this context, the present study not only reports general accuracy scores but also conducts a detailed examination of the common substitution errors and special character recognition errors produced by six different OCR models EasyOCR, Tesseract OCR, Keras OCR, DoCTR, TrOCR, and MGP-STR at the character level. These errors were computed based on each model's outputs across eight benchmark datasets, revealing distinct behavioral patterns and tendencies. Such analyses go beyond mere performance measurement, enabling a clearer understanding of which character groups each architecture struggles with and providing valuable insights for designing future improvement strategies along these dimensions.

Table 18 presents the common character confusions and special character errors identified for the EasyOCR model across

eight different datasets.

Table 18. Common substitution errors and special character recognition errors observed for the EasyOCR model across eight datasets

EasyOCR								
Metric	Cute80	ICDAR 2003	ICDAR 2013	ICDAR 2015	IIIT5K	SVT	SVTP	TotalText
Common Substitution Errors	d -> o: 3	o -> 0: 7	e -> f: 17	i -> l: 15	o -> 0: 15	i -> l: 5	o -> 0: 6	o -> 0: 15
	e -> l: 2	t -> i: 5	y -> v: 4	e -> o: 9	t -> i: 11	g -> c: 5	t -> i: 4	g -> c: 8
	-> v: 2	n -> m: 5	n -> m: 4	e -> c: 7	i -> l: 10	a -> h: 4	n -> d: 3	o -> c: 7
	n -> v: 2	e -> f: 4	r -> p: 4	s -> g: 6	e -> c: 10	e -> c: 4	i -> m: 3	i -> l: 6
	e -> y: 2	c -> g: 3	! -> i: 3	t -> l: 5	s -> \$: 9	i -> t: 3	a -> n: 3	n -> m: 6
	g -> c: 2	i -> l: 3	t -> i: 3	y -> v: 5	g -> c: 7	t -> i: 3	f -> e: 3	t -> i: 6
	t -> e: 1	t -> l: 3	d -> o: 3	o -> q: 5	t -> l: 7	c -> g: 2	g -> c: 3	o -> a: 5
	p -> i: 1	c -> d: 3	b -> 8: 3	o -> 0: 5	y -> v: 7	j -> t: 2	n -> u: 2	e -> c: 5
	n -> o: 1	i -> t: 3	i -> l: 2	f -> l: 4	r -> n: 6	l -> c: 2	a -> u: 2	p -> r: 5
	g -> o: 1	u -> 0: 2	c -> l: 2	! -> l: 4	h -> k: 6	o -> q: 2	n -> m: 2	y -> v: 4
	15	40	80	137	2	0	7	92
Total Special Characters	15	40	80	137	2	0	7	92
Special Character Errors	4	19	9	79	246	13	16	131

Across the eight scene text datasets analyzed, the EasyOCR model exhibited a strong tendency to confuse certain characters. The most prominent substitution observed was “o” → “0”, which appeared consistently across nearly all datasets (including ICDAR 2003, IIIT5K, SVTP, and TotalText). This pattern indicates that the visual similarity between the letter “o” and the numeral “0” leads to systematic misclassification in classical CNN–RNN–based architectures such as EasyOCR.

Additionally, substitutions such as “i” → “l” and “t” → “i” were frequently observed across multiple datasets. These types of errors are typically associated with factors such as low image resolution, font variation, and thin character strokes. Such confusions were particularly pronounced in more challenging datasets, including SVT, SVTP, and ICDAR 2015. Similarly, transformations like “g” → “c”, “e” → “c”, and “n” → “m” occurred due to the structural resemblance between characters and the model’s limited ability to leverage contextual linguistic information effectively.

In terms of Special Character Errors, EasyOCR demonstrated significant weaknesses. For instance, in the IIIT5K dataset, despite containing only two unique special characters, the model produced 246 special character errors. This suggests a systematic failure in recognizing or differentiating special characters, often either ignoring them or misclassifying them as other symbols. Similar issues persisted across other datasets, including TotalText (131 errors), ICDAR 2015 (79 errors), and ICDAR 2003 (19 errors), underscoring the model’s consistent limitations in handling special character recognition.

In conclusion, the EasyOCR model frequently confuses character pairs with structural similarities in character-level recognition tasks and tends to severely neglect special characters. This indicates that the model fails to adequately integrate both visual and linguistic context, placing it at a disadvantage compared to more modern Transformer-based architectures.

Table 19 presents a detailed overview of the common substitution errors and special character recognition errors observed for the Tesseract OCR model across eight different datasets.

The Tesseract OCR model exhibits a high tendency for character confusion in scene text recognition tasks. The most frequent substitution errors include “t” → “r”, “n” → “r”, “o” → “g”, “e” → “c”, “c” → “g”, and “r” → “t”. These errors indicate that the model struggles to capture discriminative visual features, particularly among rounded or open-ended characters, due to structural similarities. The frequency of such confusions increases substantially in high-variance and low-contrast datasets such as IIIT5K and ICDAR 2015.

Table 19. Common substitution errors and special character recognition errors observed for the Tesseract OCR model across eight datasets

TesseractOCR								
Metric	Cute80	ICDAR 2003	ICDAR 2013	ICDAR 2015	IIIT5K	SVT	SVTP	TotalText
Common Substitution Errors	c -> g: 4	t -> r: 3	t -> i: 4	n -> r: 8	t -> r: 10	t -> e: 3	c -> r: 3	n -> y: 8
	o -> e: 4	e -> c: 3	l -> i: 3	b -> p: 8	e -> y: 6	o -> a: 3	t -> g: 2	g -> e: 6
	t -> i: 3	s -> c: 3	o -> g: 2	p -> r: 7	i -> l: 5	h -> j: 3	r -> a: 2	l -> t: 6
	b -> p: 3	6 -> o: 2	' -> ' : 2	o -> e: 7	c -> g: 5	s -> c: 2	o -> g: 2	t -> y: 5
	a -> p: 3	0 -> u: 2	l -> !: 2	t -> i: 6	g -> s: 4	r -> e: 2	t -> l: 2	a -> 4: 5
	s -> g: 2	o -> g: 2	o -> a: 2	a -> o: 6	c -> o: 4	r -> v: 2	a -> i: 2	t -> a: 5
	t -> j: 2	c -> a: 2	d -> a: 2	e -> a: 6	r -> n: 4	a -> o: 2	u -> l: 2	i -> e: 5
	m -> n: 2	i -> a: 2	r -> f: 2	c -> e: 5	a -> o: 4	l -> t: 2	m -> n: 2	d -> e: 5
	h -> w: 2	f -> i: 2	m -> v: 2	m -> n: 5	l -> t: 4	f -> t: 1	g -> c: 2	d -> p: 5
	i -> l: 2	o -> e: 2	n -> r: 2	r -> t: 5	e -> é: 4	r -> t: 1	c -> e: 2	u -> o: 4
Total Special Characters	15	40	80	137	2	0	7	92
Special Character Errors:	35	26	25	117	305	34	24	180

The model also frequently confuses letters with simple linear distinctions (e.g., “l”, “i”, “t”), highlighting the limitations of its segmentation-based architecture. Since Tesseract OCR relies heavily on explicit segmentation, its ability to leverage contextual relationships within text is weak. Consequently, when faced with visually similar characters, the model’s error rate increases significantly.

In terms of Special Character Errors, Tesseract OCR demonstrates consistently high error rates across nearly all datasets. For instance, in the IIIT5K dataset despite containing only two special characters 305 special character errors were recorded. This indicates a systematic failure to recognize or correctly classify such symbols, often ignoring or misinterpreting them entirely. Similar weaknesses were observed in TotalText (180 errors), ICDAR 2015 (117 errors), and CUTE80 (35 errors). These findings are directly linked to the model’s strict dependence on a predefined character set and its lack of contextual flexibility.

Table 20 summarizes the common substitution errors and special character recognition errors identified for the Keras OCR model across eight datasets.

Table 20. Common substitution errors and special character recognition errors observed for the Keras OCR model across eight datasets

KerasOCR								
Metric	Cute80	ICDAR 2003	ICDAR 2013	ICDAR 2015	IIIT5K	SVT	SVTP	TotalText
Common Substitution Errors	e -> s: 6	3 -> s: 7	0 -> o: 9	r -> n: 28	i -> l: 19	i -> l: 8	m -> n: 9	m -> n: 16
	w -> n: 4	5 -> s: 5	4 -> a: 7	a -> o: 19	0 -> o: 17	f -> e: 7	t -> i: 7	e -> s: 14
	i -> t: 3	i -> l: 5	r -> s: 6	i -> l: 17	t -> l: 10	e -> c: 5	r -> n: 6	0 -> o: 12
	0 -> o: 3	r -> n: 5	p -> d: 4	t -> i: 16	r -> n: 9	t -> l: 4	a -> e: 6	5 -> s: 12
	2 -> z: 3	e -> o: 4	t -> l: 4	m -> n: 15	m -> n: 9	t -> i: 4	t -> n: 5	4 -> a: 10
	i -> a: 2	a -> e: 4	n -> m: 4	e -> c: 13	t -> i: 8	h -> n: 4	f -> e: 5	i -> l: 10
	l -> e: 2	h -> i: 4	i -> l: 3	h -> n: 12	e -> s: 8	r -> n: 4	g -> c: 4	e -> c: 10
	c -> e: 2	t -> l: 4	a -> i: 3	0 -> o: 11	a -> o: 7	t -> e: 3	i -> a: 4	a -> s: 9
	s -> l: 2	l -> i: 4	n -> r: 3	t -> l: 10	a -> e: 7	t -> c: 3	a -> n: 4	2 -> z: 9
	-> l: 2	l -> s: 4	d -> a: 3	e -> i: 10	i -> t: 7	o -> a: 3	a -> i: 4	i -> e: 9
Total Special Characters	15	40	80	137	2	0	7	92
Special Character Errors:	0	0	0	0	0	0	0	0

The Keras OCR model exhibits a distinctive pattern in its character-level recognition performance. Error analyses conducted across eight different datasets reveal that the most frequent substitution errors include “i” → “l”, “r” → “n”, “0” → “o”, “t” → “i”, “e” → “s”, and “m” → “n”. These substitutions are largely driven by visual similarity, often occurring between thin-stroked or short characters. For example, the vertical alignment of “i” and “l” makes it difficult for the model to differentiate between them in low-resolution images, resulting in recurrent misclassifications.

Additionally, substitutions such as “e” → “s”, “a” → “o”, “5” → “s”, and “2” → “z” frequently observed across multiple datasets indicate that the model experiences confusion not only between alphabetic characters but also between letters and numerals. This behavior suggests that Keras OCR relies predominantly on visual pattern recognition and makes limited use of linguistic contextual cues during decoding.

Notably, the Keras OCR model did not produce any special character errors. Across datasets with varying levels of text complexity such as CUTE80, ICDAR, IIIT5K, SVT, SVTP, and TotalText the model consistently demonstrated strong performance in recognizing special characters.

Table 21 presents the common substitution errors and special character recognition errors identified for the DoCTR model across eight datasets.

Table 21. Common substitution errors and special character recognition errors observed for the DoCTR model across eight datasets

DoCTR								
Metric	Cute80	ICDAR 2003	ICDAR 2013	ICDAR 2015	IIIT5K	SVT	SVTP	TotalText
Common Substitution Errors	l -> a: 3	o -> 0: 7	b -> 8: 4	e -> a: 13	i -> l: 13	i -> t: 5	t -> i: 6	s -> a: 13
	o -> u: 3	g -> c: 6	e -> o: 3	a -> o: 12	e -> s: 8	t -> l: 4	m -> n: 5	o -> a: 13
	d -> a: 3	-> c: 4	i -> f: 3	i -> l: 10	t -> l: 8	o -> u: 4	a -> o: 5	r -> a: 12
	c -> e: 3	> t: 3	g -> t: 3	s -> a: 10	c -> e: 7	r -> a: 3	o -> 0: 5	t -> i: 9
	e -> s: 3	-> a: 3	r -> i: 3	e -> i: 9	i -> u: 7	t -> i: 3	o -> a: 4	e -> a: 9
	r -> a: 2	p -> r: 3	o -> 0: 3	a -> i: 9	c -> g: 6	h -> l: 3	f -> e: 4	t -> a: 9
	s -> .: 2	r -> i: 2	p -> i: 2	o -> a: 9	o -> a: 6	c -> e: 3	c -> g: 4	h -> i: 9
	m -> l: 2	e -> o: 2	m -> h: 2	m -> n: 9	f -> e: 6	e -> l: 2	t -> l: 4	l -> i: 8
	h -> m: 2	i -> o: 2	! -> i: 2	o -> u: 9	o -> u: 6	t -> a: 2	g -> c: 3	n -> s: 8
	s -> n: 2	h -> a: 2	j -> i: 2	i -> t: 9	l -> i: 6	a -> n: 2	a -> u: 3	m -> n: 8
Total Special Characters	15	40	80	137	2	0	7	92
Special Character Errors:	34	30	29	39	358	16	8	203

Despite its Transformer-based architecture, the DoCTR model exhibits noticeable character confusion patterns, particularly in scene texts containing low-resolution or stylized writing. The most frequently observed substitution errors include “s” → “a”, “t” → “i”, “i” → “l”, “o” → “a”, “e” → “a”, “m” → “n”, “l” → “a”, and “o” → “u”. These substitutions suggest that the model occasionally struggles to distinguish between visually similar character forms.

Such errors are especially prominent in complex datasets like ICDAR 2015, IIIT5K, and TotalText. For instance, in IIIT5K, frequent substitutions such as “e” → “a”, “a” → “o”, “i” → “l”, “t” → “l”, and “c” → “e” were observed in large numbers. This indicates that the model’s spatial attention mechanism may not be optimally leveraged when processing long or naturally aligned scene text sequences.

In terms of Special Character Errors, DoCTR displayed notably high error rates in certain datasets particularly IIIT5K and TotalText. Although IIIT5K contains only two unique special characters, the model produced 358 errors, while 203 errors were recorded for TotalText, indicating substantial deficiencies in recognizing special characters. This weakness may be attributed to insufficient representation of such symbols in the training data or to the model’s tendency to interpret them as visual noise.

Conversely, in relatively simpler or special-character-free datasets such as SVT and SVTP, the number of errors was significantly lower. This pattern suggests that the model’s recognition capability is sensitive to the dataset structure, performing more reliably when the visual variability of special characters is reduced.

Table 22 presents the common substitution errors and special character recognition errors identified for the TrOCR model across eight datasets.

Table 22. Common substitution errors and special character recognition errors observed for the TrOCR model across eight datasets

TrOCR								
Metric	Cute80	ICDAR 2003	ICDAR 2013	ICDAR 2015	IIIT5K	SVT	SVTP	TotalText
Common Substitution Errors	c -> o: 6	o -> 0:17	l -> t: 14	a -> o: 42	o -> 0: 33	a -> o: 15	a -> e: 18	o -> 0: 34
	o -> 0: 6	a -> e: 13	n -> m:12	a -> e: 38	a -> e: 33	m -> n:14	a -> o: 18	e -> i: 32
	c -> s: 5	a -> i: 8	e -> i: 10	a -> i: 38	a -> o: 29	a -> i: 12	r -> n: 13	a -> e: 27
	m -> n: 5	t -> n: 8	g -> c: 8	s -> 0: 36	e -> i: 27	a -> e: 12	c -> 0: 13	a -> i: 25
	t -> s: 5	g -> c: 7	c -> 0: 7	r -> n: 35	n -> m:26	c -> o: 9	r -> t: 13	o -> e: 23
	o -> .: 5	r -> t: 7	d -> p: 7	a -> 0:30	h -> i: 24	n -> t: 9	e -> i: 11	h -> i: 22
	o -> s: 4	c -> o: 7	l -> i: 7	t -> l: 24	r -> n: 23	e -> o: 7	h -> p:10	c -> 0: 21
	a -> e: 4	r -> l: 6	a -> e: 7	e -> i: 24	m -> n:22	a -> n: 7	l -> n: 10	a -> o: 20
	r -> p: 4	g -> o: 6	a -> 0: 7	o -> e: 23	c -> o: 19	l -> n: 7	m -> 0: 9	a -> s: 20
	a -> y: 4	c -> 0: 6	c -> o: 6	o -> 0:23	t -> i: 19	l -> i: 7	n -> t: 9	s -> 0: 20
Total Special Characters	15	40	80	137	2	0	7	92
Special Character Errors:	69	178	113	439	907	170	153	528

Although the TrOCR model integrates Vision Transformer components with strong linguistic context modeling, it continues to face significant challenges in accurately distinguishing characters within scene text recognition tasks. The data presented in the table indicate that the most frequent substitution errors include “o” → “0”, “a” → “o”, “a” → “e”, “e” → “i”, “a” → “i”, “r” → “n”, “m” → “n”, and “c” → “o/0”. These errors highlight a recurring confusion between letters and digits as well as between characters with similar anatomical structures, revealing persistent visual ambiguity issues.

The TrOCR model generated notably high error counts, particularly on the ICDAR 2015, IIIT5K, and TotalText datasets. For instance, in IIIT5K, transformations such as “a” → “o”, “o” → “0”, and “a” → “e” occurred more than 30 times each, while in TotalText, the single substitution “o” → “0” appeared 34 times. Despite being among the most advanced architectures, the model’s repeated misclassifications demonstrate its sensitivity to shape-based similarities in scene texts.

In terms of Special Character Errors, TrOCR produced the highest number of errors among all models. In the IIIT5K dataset, only two special characters resulted in 907 misclassifications, while 528 errors were observed in TotalText. Even in simpler datasets such as SVT and SVTP, the model generated over 150 special character errors, indicating an excessive sensitivity that leads to frequent misassignments of special characters.

Table 23 presents the common substitution errors and special character recognition errors observed for the MGP-STR model across eight scene text datasets.

Table 23. Common substitution errors and special character recognition errors observed for the MGP-STR model across eight datasets

MGP-STR								
Metric	Cute80	ICDAR 2003	ICDAR 2013	ICDAR 2015	IIIT5K	SVT	SVTP	TotalText
Common Substitution Errors	b -> t: 3	& -> 8: 4	0 -> w: 3	o -> a: 7	t -> o: 3	a -> u: 2	g -> i: 3	a -> i: 7
	r -> n: 3	l -> i: 4	b -> 8: 2	a -> o: 5	m -> o: 3	i -> t: 1	é -> e: 2	a -> o: 6
	s -> e: 2	& -> a: 3	3 -> i: 1	i -> e: 4	n -> f: 3	c -> r: 1	a -> o: 2	i -> h: 6
	-> h: 2	f -> i: 3	i -> t: 1	o -> i: 4	a -> u: 3	b -> e: 1	r -> t: 2	t -> i: 6
	. -> o: 2	i -> e: 2	o -> d: 1	e -> a: 4	r -> e: 3	o -> f: 1	o -> 0: 2	t -> n: 6
	u -> o: 2	e -> a: 2	u -> i: 1	t -> l: 4	o -> p: 2	z -> v: 1	m -> i: 2	t -> r: 5
	t -> o: 2	h -> i: 2	g -> c: 1	l -> p: 4	o -> d: 2	g -> c: 1	f -> r: 2	s -> t: 5
	-> t: 1	i -> l: 2	s -> u: 1	i -> l: 4	7 -> 2: 2	t -> r: 1	h -> r: 1	e -> o: 5
	-> s: 1	0 -> o: 2	e -> d: 1	f -> a: 3	i -> l: 2	s -> i: 1	a -> u: 1	c -> g: 4
	n -> h: 1	£ -> e: 2	a -> i: 1	o -> n: 3	l -> p: 2	c -> a: 1	t -> a: 1	l -> t: 4
Total Special Characters	15	40	80	137	2	0	7	92
Special Character Errors:	0	0	0	2	0	0	0	0

The MGP-STR model, a Vision Transformer-based advanced architecture distinguished by its high accuracy in scene text recognition, also demonstrates superior performance in character recognition tasks. As shown in Table 23, the model’s

character substitution patterns are not only few in number but also limited in diversity. The most common substitution errors include “a” → “i/o”, “o” → “a/i”, “i” → “e/l”, “t” → “r/n”, and “r” → “n/t”. These confusions generally occur between character pairs with very subtle visual distinctions and remain remarkably infrequent despite the challenging conditions of scene text environments.

When compared with models that produced the highest number of errors, such as TrOCR and DoCTR, the frequency of errors in MGP-STR is exceptionally low. For instance, while the TrOCR model generated over 900 special character errors in the IIIT5K dataset, MGP-STR recorded only two. Moreover, in the remaining seven datasets, no special character errors were observed. This outcome demonstrates the model’s high precision in both character segmentation and classification, as well as its robust capability to accurately recognize and filter special characters.

In conclusion, thanks to its multi-granularity architecture, which integrates multi-scale information flow with linguistic context, the MGP-STR model delivers outstanding overall performance. The conducted error type analyses reveal that the structural differences among OCR models influence not only their accuracy rates but also their error diversity and special character handling capabilities. Notably, models such as MGP-STR and Keras OCR exhibited low error frequencies, whereas TrOCR and DoCTR produced high error densities and systematic character confusions.

4.4. MGP-STR Model Performance on the Turkish Scene Text Dataset

In this section, unlike the previous comparative analyses, the MGP-STR model’s performance is evaluated specifically on a Turkish Scene Text Dataset, which contains scene text samples in the Turkish language. The analysis focuses on both character-level and word-level recognition performance. The test set comprises 2,079 words [68]. Additionally, the impact of Turkish-specific characters and the morphological structure of the language on model performance is discussed.

Table 24 presents the character-level and word-level performance results of the MGP-STR model on the TS-TR (Turkish Scene Text) dataset.

Table 24. Character- and word level performance results of the MGP-STR model on the TS-TR dataset

TS-TR: Turkish scene text dataset		
Models		Mgp-STR
Metric		
Character	Accuracy	0.8541
	F1 Score	0.8956
	Recall	0.8695
	Precision	0.9234
	Levenshtein	1880
	Total / Correct	12880 / 11001
Word	Accuracy	0.4978
	F1 Score	0.4978
	Recall	0.4978
	Precision	0.4978
	Total / Correct	2079/1035

Although the MGP-STR model has demonstrated outstanding performance on English-language scene text datasets, its results on the TS-TR (Turkish Scene Text) dataset clearly reveal the influence of linguistic differences on model performance. At the character level, the model achieved an accuracy of 85.41%, an F1-score of 89.56%, and a Levenshtein distance of 1880, indicating generally strong recognition capability. However, this success declines substantially when evaluated at the word level.

All word-level metrics including accuracy, F1-score, precision, and recall were measured at only 49.78%, meaning the model correctly recognized only one out of every two words in the Turkish dataset. Compared to the 77–97% accuracy range observed in English scene text benchmarks, this represents a significant performance drop. These results indicate that the model struggles with Turkish-specific morphological structures, character combinations, and special letters, suggesting a limited capacity for cross-lingual generalization and language adaptation.

The most frequent character substitutions observed in the Turkish dataset include: “ü” → “u” (168 times), “ş” → “s” (154 times), “ç” → “c” (113 times), “ğ” → “g” (87 times), “ö” → “o” (60 times), and cases where the dot above “i” was misread, leading to substitutions such as “i” → “s” (29 times) or “i” → “c” (16 times). Additionally, less frequent confusions such as “ç” → “g” (9 times) and “y” → “v” (6 times) were also observed. These substitution patterns indicate that the model fails to accurately distinguish Turkish-specific diacritical characters, instead mapping them to their default Latin counterparts. Such systematic transformations play a major role in explaining the drastic reduction in word-level performance. The confusion of Turkish characters with visually similar Latin letters, coupled with insufficient modeling of linguistic context, contributes significantly to these substitution errors.

5. Discussion

In this study, six OCR models were comparatively analyzed across eight English scene text datasets and one TS-TR. The findings demonstrate that model architecture plays a decisive role not only in determining overall accuracy but also in shaping character-level error patterns, special character recognition performance, and language adaptability.

The Transformer-based MGP-STR model achieved the highest overall accuracy at both the character and word levels, exhibiting superior performance particularly on English scene text datasets. However, when applied to the Turkish scene text dataset containing language-specific characters and morphological complexity, the model’s performance declined sharply at the word level (49.78% accuracy). This significant drop underscores the model’s limited language adaptation capability and its difficulty in accurately recognizing Turkish-specific diacritics and orthographic forms.

Despite having large parameter counts and complex architectures, Transformer-based models such as TrOCR and DoCTR exhibited systematic character confusions and particularly high rates of special character errors across multiple datasets. For instance, in the IIIT5K dataset, the TrOCR model produced 907 special character errors, even though the dataset contained only two unique special characters. This finding indicates that complex architectures do not necessarily guarantee superior overall performance and that data diversity and language representation are just as critical as architectural sophistication.

Traditional rule-based systems such as Tesseract OCR, which rely heavily on explicit character segmentation, demonstrated lower accuracy and higher error rates compared to modern deep learning models. The absence of contextual understanding made these systems inadequate when dealing with complex or visually distorted scene texts. Mid-level deep learning frameworks such as Keras OCR and EasyOCR delivered relatively stable results in certain datasets; however, their performance declined in multi-oriented or heavily distorted scenes due to their limited capability for contextual reasoning and linguistic integration.

In light of these findings, it becomes evident that OCR systems should be evaluated not solely based on visual accuracy, but through multi-dimensional criteria encompassing character substitution patterns, special character sensitivity, word-level contextual performance, and language adaptability. For systems targeting low-resource languages, linguistic factors such as morphological richness and unique alphabetic characters must be explicitly incorporated into both model design and evaluation processes.

The results highlight several promising directions for improving model performance. For languages containing distinctive alphabetic elements such as the Turkish characters ç, ğ, ş, ü, ö character-level language adaptation becomes essential. In this context, methods such as multilingual pre-training, linguistically informed data augmentation, character-specific tokenization, and visual-linguistic hybrid architectures are strongly recommended. Furthermore, to reduce word-level recognition errors, the integration of language models, post-processing correction mechanisms, and context-aware attention layers should be considered to enhance both contextual understanding and textual consistency.

The models examined in this study were evaluated in their pre-trained forms, each developed with different training datasets, hyperparameter settings, and optimization strategies. This introduces certain limitations when making direct

architectural comparisons. Furthermore, the first version of TS-TR dataset has a relatively limited scope, particularly in terms of font diversity, scene variability, and natural distortions, indicating significant potential for further expansion and refinement in future work.

6. Conclusion and Future Work

In this study, six different OCR models were comparatively evaluated at both the character and word levels across eight English and one Turkish scene text datasets. The MGP-STR model achieved the highest overall accuracy; however, it exhibited a marked performance decline at the word level on the TS-TR dataset, which includes Turkish-specific characters. The findings highlight the need for further improvement of OCR systems in terms of language awareness, special character processing, and contextual generalization capabilities. In this regard, multilingual training strategies and language-specific architectural enhancements should be prioritized in future research.

Future studies should focus on developing larger, multi-scene, and linguistically rich Turkish datasets, implementing transfer learning-based Turkish model adaptation, and conducting semantic-level analyses of character substitution errors to accelerate progress in this field. Moreover, next-generation OCR architectures that jointly encode linguistic and visual context hold strong potential for delivering more effective solutions for Turkish and other languages with similar morphological complexity.

Conflict of Interest Statement

The authors declare that there is no conflict of interest

References

- [1] J. Baek et al., "What is wrong with scene text recognition model comparisons? Dataset and model analysis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV 2019)*, October 27–November 2, 2019, Seoul, Korea [Online]. Available: IEEE/CVF Open Access, https://openaccess.thecvf.com/content_ICCV_2019/ [Accessed: 29 Dec. 2025].
- [2] R. Smith, "An overview of the Tesseract OCR engine," in *Proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, September 23–26, 2007, Curitiba, Brazil [Online]. Available: IEEE Xplore, <https://ieeexplore.ieee.org/>. [Accessed: 29 Dec. 2025].
- [3] B. Shi, X. Wang, P. Lyu, C. Yao, and X. Bai, "Robust scene text recognition with automatic rectification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016)*, June 26–July 1, 2016, Las Vegas, NV, USA [Online]. Available: CVF Open Access, <https://www.cv-foundation.org/>. [Accessed: 29 Dec. 2025].
- [4] Y. L. Tan, A. W. K. Kong, and J. J. Kim, "Pure transformer with integrated experts for scene text recognition," in *Computer Vision – ECCV 2022: Proceedings of the 17th European Conference on Computer Vision (ECCV 2022)*, October 23–27, 2022, Tel Aviv, Israel, S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds. Cham: Springer, 2022, Lecture Notes in Computer Science, vol. 13688. pp. 486–502. doi: 10.1007/978-3-031-19815-1_28
- [5] M. Li et al., "TrOCR: Transformer-based optical character recognition with pre-trained models," in *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI 2023)*, February 7–14, 2023, B. Williams, Y. Chen, and J. Neville, Eds. Washington, DC: AAAI Press, 2023. pp. 13094–13102. doi: 10.1609/aaai.v37i11.26538
- [6] P. Wang, C. Da, and C. Yao, "Multi-granularity prediction for scene text recognition," in *Computer Vision – ECCV 2022: Proceedings of the 17th European Conference on Computer Vision (ECCV 2022)*, October 23–27, 2022, Tel Aviv, Israel, S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds. Cham: Springer, 2022, Lecture Notes in Computer Science, vol. 13688. pp. 339–355. doi: 10.1007/978-3-031-19815-1_20
- [7] X. Chen, L. Jin, Y. Zhu, C. Luo, and T. Wang, "Text recognition in the wild: A survey," *ACM Computing Surveys (CSUR)*, vol. 54, no. 2, pp. 1–35, 2021. doi: 10.1145/3440756
- [8] H. Yue et al., "NRSTRNet: a novel network for noise-robust scene text recognition," *International Journal of Computational Intelligence Systems*, vol. 16, no. 1, p. 5, 2023. doi: 10.1007/s44196-023-00181-1

- [9] J. Baek et al., "What is wrong with scene text recognition model comparisons? Dataset and model analysis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV 2019), October 27–November 2, 2019, Seoul, Korea* [Online]. Available: IEEE/CVF Open Access, https://openaccess.thecvf.com/content_ICCV_2019/ [Accessed: 29 Dec. 2025].
- [10] M. Şahin, E. Şahin, E. Özdemir, F. Talu, and S. Öztürk, "Beyin tümörü biyopsisi için derin öğrenme tabanlı risk minimizasyonlu otomatik planlama," *Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi*, vol. 40, no. 1, pp. 487–500, 2024. doi: 10.17341/gazimmfd.1348325
- [11] M. A. Dursun and P. Ö. KAVAS, "Beyond Diagnosis: Cross-Dataset Evaluation of Risk Factors for Thyroid Cancer Recurrence," *Artificial Intelligence Studies*, vol. 8, no. 1, pp. 38–69, 2025. doi: 10.30855/ais.2025.08.01.03
- [12] E. Şahin and D. Özdemir, "ThinkSTra: a transformer-driven architecture for decoding imagined speech from EEG with spatial-temporal dynamics," *Medical & Biological Engineering & Computing*, pp. 1–26, 2025. doi: 10.1007/s11517-025-03478-9
- [13] E. Şahin and M. F. Talu, "WY-NET: A new approach to image synthesis with generative adversarial networks," *Journal of Scientific Reports-A*, no. 050, pp. 270–290, 2022.
- [14] S. Dörterler, "Kanser hastalığı teşhisinde ölüm oyunu optimizasyon algoritmasının etkisi," in *Mühendislik Alanında Uluslararası Araştırmalar VIII*, N. Görür, Ed. Istanbul: Eğitim Publishing, 2023, p. 15.
- [15] İ. Şahin, M. Dörterler, and H. GÖKÇE, "Optimization of hydrostatic thrust bearing using enhanced grey wolf optimizer," *Mechanics*, vol. 25, no. 6, pp. 480–486, 2019. doi: 10.5755/j01.mech.25.6.22512
- [16] S. Dörterler, S. Arslan, and D. Özdemir, "Unlocking the potential: A review of artificial intelligence applications in wind energy," *Expert Systems*, vol. 41, no. 12, p. e13716, 2024. doi: 10.1111/exsy.13716
- [17] V. Kaya, "Classification of waste materials with a smart garbage system for sustainable development: A novel model," *Frontiers in Environmental Science*, vol. 11, p. 1228732, 2023. doi: 10.3389/fenvs.2023.1228732
- [18] N. Yagmur, I. Dag, and H. Temurtas, "A new computer-aided diagnostic method for classifying anaemia disease: Hybrid use of Tree Bagger and metaheuristics," *Expert Systems*, p. e13528, 2023. doi: 10.1111/exsy.13528
- [19] S. Dörterler, H. Dumlu, D. Özdemir, and H. Temurtas, "Melezlenmiş K-means ve diferansiyel gelişim algoritmaları ile kalp hastalığının teşhisi," in *Proceedings of the 1st International Conference on Engineering and Applied Natural Sciences (ICEANS 2022), May 10–13, 2022, Konya, Turkey*, U. Özkaya, Ed. Konya: ICEANS, 2022. pp. 1840–1844.
- [20] G. Arslan, F. Aydemir, and S. Arslan, "Enhanced license plate recognition using deep learning and block-based approach," *Journal of Scientific Reports-A*, no. 58, pp. 57–82, 2023.
- [21] G. Nagy, "Twenty years of document image analysis in PAMI," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 22, no. 01, pp. 38–62, 2000. doi: 10.1109/34.824820
- [22] D. Karatzas et al., "ICDAR 2015 competition on robust reading," in *Proceedings of the 13th International Conference on Document Analysis and Recognition (ICDAR 2015), August 23–26, 2015, Tunis, Tunisia* [Online]. Available: IEEE Xplore, <https://ieeexplore.ieee.org/>. [Accessed: 29 Dec. 2025].
- [23] X. Chen, L. Jin, Y. Zhu, C. Luo, and T. Wang, "Text recognition in the wild: A survey," *ACM Computing Surveys (CSUR)*, vol. 54, no. 2, pp. 1–35, 2021. doi: 10.1145/3440756
- [24] K. Wang, B. Babenko, and S. Belongie, "End-to-end scene text recognition," in *Proceedings of the 2011 International Conference on Computer Vision (ICCV 2011), November 6–13, 2011, Barcelona, Spain* [Online]. Available: IEEE Xplore, <https://ieeexplore.ieee.org/>. [Accessed: 29 Dec. 2025]. doi: 10.1109/ICCV.2011.6126402
- [25] S. He, R. W. H. Lau, W. Liu, Z. Huang, and Q. Yang, "Supercnn: A superpixelwise convolutional neural network for salient object detection," *International Journal of Computer Vision*, vol. 115, pp. 330–344, 2015. doi: 10.1007/s11263-015-0822-0
- [26] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, "Character region awareness for text detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2019), June 16–20, 2019, Long Beach, CA, USA* [Online]. Available: CVF Open Access, <https://openaccess.thecvf.com/>. [Accessed: 29 Dec. 2025].
- [27] A. Bissacco, M. Cummins, Y. Netzer, and H. Neven, "PhotoOCR: Reading text in uncontrolled conditions," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV 2013), December 1–8, 2013, Sydney, Australia* [Online]. Available: CVF Open Access, <https://openaccess.thecvf.com/>. [Accessed: 29 Dec. 2025].
- [28] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 11, pp. 2298–2304, 2016. doi: 10.1109/TPAMI.2016.2646371

- [29] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Synthetic data and artificial neural networks for natural scene text recognition," *arXiv preprint arXiv:1406.2227*, 2014. doi: 10.48550/arXiv.1406.2227
- [30] A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic data for text localisation in natural images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016), June 26–July 1, 2016, Las Vegas, NV, USA* [Online]. Available: CVF Open Access, <https://openaccess.thecvf.com/>. [Accessed: 29 Dec. 2025].
- [31] S. Long and C. Yao, "Unrealtext: Synthesizing realistic scene text images from the unreal world," *arXiv preprint arXiv:2003.10608*, 2020. doi: 10.48550/arXiv.2003.10608
- [32] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, "Character region awareness for text detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2019), June 16–20, 2019, Long Beach, CA, USA* [Online]. Available: CVF Open Access, <https://openaccess.thecvf.com/>. [Accessed: 29 Dec. 2025].
- [33] C.-Y. Lee and S. Osindero, "Recursive recurrent nets with attention modeling for OCR in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016), June 26–July 1, 2016, Las Vegas, NV, USA* [Online]. Available: CVF Open Access, <https://openaccess.thecvf.com/>. [Accessed: 29 Dec. 2025].
- [34] B. Shi, M. Yang, X. Wang, P. Lyu, C. Yao, and X. Bai, "Aster: An attentional scene text recognizer with flexible rectification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 9, pp. 2035–2048, 2018. doi: 10.1109/TPAMI.2018.2848939
- [35] B. Shi, X. Wang, P. Lyu, C. Yao, and X. Bai, "Robust scene text recognition with automatic rectification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016), June 26–July 1, 2016, Las Vegas, NV, USA* [Online]. Available: CVF Open Access, <https://www.cv-foundation.org/>. [Accessed: 29 Dec. 2025].
- [36] Z. Cheng, Y. Xu, F. Bai, Y. Niu, S. Pu, and S. Zhou, "AON: Towards arbitrarily-oriented text recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2018), June 18–22, 2018, Salt Lake City, UT, USA* [Online]. Available: CVF Open Access, <https://openaccess.thecvf.com/>. [Accessed: 29 Dec. 2025].
- [37] Y. Baek et al., "CLEval: Character-level evaluation for text detection and recognition tasks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW 2020), June 14–19, 2020, Virtual Conference* [Online]. Available: CVF Open Access, <https://openaccess.thecvf.com/>. [Accessed: 29 Dec. 2025].
- [38] J. I. Laszlo and P. J. Bairstow, "Handwriting: Difficulties and possible solutions," *School Psychology International*, vol. 5, no. 4, pp. 207–213, 1984. doi: 10.1177/0143034384054004
- [39] H. Bunke, M. Roth, and E. G. Schukat-Talamazzini, "Off-line cursive handwriting recognition using hidden markov models," *Pattern Recognition*, vol. 28, no. 9, pp. 1399–1413, 1995. doi: 10.1016/0031-3203(95)00013-P
- [40] W. Cho, S.-W. Lee, and J. H. Kim, "Modeling and recognition of cursive words with hidden Markov models," *Pattern Recognition*, vol. 28, no. 12, pp. 1941–1953, 1995. doi: 10.1016/0031-3203(95)00041-0
- [41] A. Kundu, Y. He, and P. Bahl, "Recognition of handwritten word: first and second order hidden Markov model based approach," *Pattern Recognition*, vol. 22, no. 3, pp. 283–297, 1989. doi: 10.1016/0031-3203(89)90076-9
- [42] M. Gilloux, "Hidden Markov models in handwriting recognition," in *Fundamentals in Handwriting Recognition, Proceedings of the NATO Advanced Study Institute on Handwriting Recognition*, S. Impedovo, Ed. Berlin, Heidelberg: Springer, 1994. pp. 264–288.
- [43] J. Hu, M. K. Brown, and W. Turin, "HMM based online handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 10, pp. 1039–1045, 1996. doi: 10.1109/34.541414
- [44] W. AlKendi, F. Gechter, L. Heyberger, and C. Guyeux, "Advancements and challenges in handwritten text recognition: A comprehensive survey," *Journal of Imaging*, vol. 10, no. 1, p. 18, 2024. doi: 10.3390/jimaging10010018
- [45] K. Barrere, Y. Soullard, A. Lemaitre, and B. Coüasnon, "A light transformer-based architecture for handwritten text recognition," in *Document Analysis Systems: Proceedings of the 15th IAPR International Workshop (DAS 2022), May 22–25, 2022, La Rochelle, France*, S. Uchida, E. Barney, and V. Eglin, Eds. Cham: Springer, 2022, Lecture Notes in Computer Science, vol. 13237. pp. 275–290.
- [46] H. Singh, R. K. Sharma, and V. P. Singh, "Online handwriting recognition systems for Indic and non-Indic scripts: A review," *Artificial Intelligence Review*, vol. 54, no. 2, pp. 1525–1579, 2021. doi: 10.1007/s10462-020-09886-7
- [47] S. S. Rosyda and T. W. Purboyo, "A review of various handwriting recognition methods," *International Journal of Applied Engineering Research*, vol. 13, no. 2, pp. 1155–1164, 2018.

- [48] P. Yadav and N. Yadav, "Handwriting recognition system-a review," *International Journal of Computer Applications*, vol. 114, no. 19, pp. 36–40, 2015.
- [49] S. Singh, S. Rohilla, and A. Sharma, "An inclusive review on deep learning techniques and their scope in handwriting recognition," *arXiv preprint arXiv:2404.08011*, 2024. doi: 10.48550/arXiv.2404.08011
- [50] M. Agarwal and A. Anastasopoulos, "A concise survey of OCR for low-resource languages," in *Proceedings of the 4th Workshop on Natural Language Processing for Indigenous Languages of the Americas (AmericasNLP 2024)*, June 2024, Mexico City, Mexico [Online]. Available: ACL Anthology, <https://aclanthology.org/>. [Accessed: 29 Dec. 2025]. doi: 10.18653/v1/2024.americasnlp-1.10
- [51] M. Xu, J. Zhang, L. Xu, W. Silamu, and Y. Li, "Collaborative encoding method for scene text recognition in low linguistic resources: The Uyghur language case study," *Applied Sciences*, vol. 14, no. 5, p. 1707, 2024. doi: 10.3390/app14051707
- [52] M. Nazeem, R. Anitha, and S. Navaneeth, "Open-source OCR libraries: A comprehensive study for low resource language," in *Proceedings of the 21st International Conference on Natural Language Processing (ICON 2024)*, December 2024, Chennai, India [Online]. Available: ACL Anthology, <https://aclanthology.org/>. [Accessed: 29 Dec. 2025].
- [53] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, "Character region awareness for text detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2019)*, June 16–20, 2019, Long Beach, CA, USA [Online]. Available: CVF Open Access, <https://openaccess.thecvf.com/>. [Accessed: 29 Dec. 2025].
- [54] Jaied, "EasyOCR," Dec. 14, 2025. [Online]. Available: <https://github.com/JaiedAI/EasyOCR>. [Accessed: Dec. 14, 2025].
- [55] P. Wang, C. Da, and C. Yao, "Multi-granularity prediction for scene text recognition," in *Computer Vision – ECCV 2022: Proceedings of the 17th European Conference on Computer Vision (ECCV 2022)*, October 23–27, 2022, Tel Aviv, Israel, S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds. Cham: Springer, 2022, Lecture Notes in Computer Science, vol. 13688. pp. 339–355. doi: 10.1007/978-3-031-19815-1_20
- [56] R. Smith, "An overview of the Tesseract OCR engine," in *Proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, September 23–26, 2007, Curitiba, Brazil [Online]. Available: IEEE Xplore, <https://ieeexplore.ieee.org/>. [Accessed: 29 Dec. 2025].
- [57] D. Sporic, E. Cuşnir, and C.-A. Boiangiu, "Improving the accuracy of Tesseract 4.0 OCR engine using convolution-based preprocessing," *Symmetry*, vol. 12, no. 5, p. 715, 2020. doi: 10.3390/sym12050715
- [58] D. Mindee, "Document text recognition." Dec. 14, 2025. [Online]. Available: <https://github.com/mindee/doctr>. [Accessed: Dec. 14, 2025].
- [59] M. Liao, Z. Wan, C. Yao, K. Chen, and X. Bai, "Real-time scene text detection with differentiable binarization," in *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI 2020)*, February 7–12, 2020, New York, NY, USA. Palo Alto, CA: AAAI Press, 2020. pp. 11474–11481.
- [60] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu, "Detecting texts of arbitrary orientations in natural images," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2012)*, June 16–21, 2012, Providence, RI, USA [Online]. Available: IEEE Xplore, <https://ieeexplore.ieee.org/>. [Accessed: 29 Dec. 2025].
- [61] S. M. Lucas et al., "ICDAR 2003 robust reading competitions: entries, results, and future directions," *International Journal of Document Analysis and Recognition (IJDR)*, vol. 7, pp. 105–122, 2005. doi: 10.1007/s10032-004-0134-3
- [62] D. Karatzas et al., "ICDAR 2013 robust reading competition," in *Proceedings of the 12th International Conference on Document Analysis and Recognition (ICDAR 2013)*, August 25–28, 2013, Washington, DC, USA [Online]. Available: IEEE Xplore, <https://ieeexplore.ieee.org/>. [Accessed: 29 Dec. 2025]. doi: 10.1109/ICDAR.2013.221
- [63] O. Y. Ling, L. B. Theng, A. C. Weiyeen, and C. McCarthy, "Development of vertical text interpreter for natural scene images," *IEEE Access*, vol. 9, pp. 144341–144351, 2021. doi: 10.1109/ACCESS.2021.3121608
- [64] J. Baek, Y. Matsui, and K. Aizawa, "What if we only use real datasets for scene text recognition? Toward scene text recognition with fewer labels," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2021)*, June 19–25, 2021, Virtual Conference [Online]. Available: CVF Open Access, <https://openaccess.thecvf.com/>. [Accessed: 29 Dec. 2025].
- [65] C. K. Ch'ng and C. S. Chan, "Total-text: A comprehensive dataset for scene text detection and recognition," in *Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition (ICDAR 2017)*, November 9–15, 2017, Kyoto, Japan [Online]. Available: IEEE Xplore, <https://ieeexplore.ieee.org/>. [Accessed: 29 Dec. 2025]. doi: 10.1109/ICDAR.2017.157
- [66] A. Risnumawan, P. Shivakumara, C. S. Chan, and C. L. Tan, "A robust arbitrary text detection system for natural scene images," *Expert Systems with Applications*, vol. 41, no. 18, pp. 8027–8048, 2014. doi: 10.1016/j.eswa.2014.07.008

[67] S. Dörterler, H. Dumlu, D. Özdemir, and H. Temurtaş, “Hybridization of meta-heuristic algorithms with K-means for clustering analysis: Case of medical datasets,” *Gazi Mühendislik Bilimleri Dergisi*, pp. 1–23. doi: 10.30855/gmbd.0705N01

[68] S. Yıldız, “Turkish scene text recognition: Introducing extensive real and synthetic datasets and a novel recognition model,” *Engineering Science and Technology, an International Journal*, vol. 60, p. 101881, 2024. doi: 10.1016/j.jestch.2024.101881

This is an open access article under the CC-BY license

