Open Access Library Journal
2025, Volume 12, e13574
ISSN Online: 2333-9721
ISSN Print: 2333-9705

Multilingual Text Recognition and Assistance for Low-Resource Languages Using Computer Vision

Franck Senu Binunya, Huabing Zhou

Hubei Key Laboratory of Intelligent Robot, Wuhan Institute of Technology, Wuhan, China
Email: franckbinunya4@gmail.com, zhouhuabing@gmail.com

How to cite this paper: Binunya, F.S. and Zhou, H.B. (2025) Multilingual Text Recognition and Assistance for Low-Resource Languages Using Computer Vision. *Open Access Library Journal*, **12**: e13574.
https://doi.org/10.4236/oalib.1113574

Received: May 8, 2025
Accepted: June 27, 2025
Published: June 30, 2025

Abstract

In this paper, a novel multilingual OCR (Optical Character Recognition) method for scanned papers is provided. Current open-source solutions, like Tesseract, offer extremely high accuracy when it comes to Latin letters. Nonetheless, multilingual texts using Asian characters typically have less accuracy than ones that are simply in Latin. The challenges for OCR increase when handling the logographic Chinese and Korean scripts because these languages feature complex multi-stroke characters. Text segmentation in these scripts proves challenging because their scripts lack word boundaries that Latin-based languages possess. OCR performs substantially worse at document processing when mixed English, Chinese, and Korean content exists within a single document rather than when the documents contain English-only content. The mix of complex character structures that includes no word boundaries and numerous dense character sets leads to subpar performance of current OCR systems, which process multilingual content. We provide a novel architecture that addresses these issues by using three neural blocks a segmenter and switcher as well as numerous recognizers as well as the segmenter's reinforcement learning: Our system solves multilingual OCR challenges by implementing a segmenter to separate word images into single-character sub-images which helps minimize the difficulties of recognizing multi-stroke characters present in Chinese and Korean languages. Each sub-image goes to the switcher, which distributes it among specialized recognizers to increase accuracy due to task assignments based on character type. A new approach deals with non-Latin script word boundaries by eliminating their identification challenges. The training process of recognizers through supervised learning enhances both character recognition performance and overall output for multilingual documents. Nevertheless, there are two significant problems with the segmenter's supervised learning: Its training necessitates a significant amount of annotation work, and its target function is not optimal.

Therefore, by using the reinforcement learning method, training for the segmenter can minimise the edit distance of the final recognition results, thereby optimising overall performance. According to experimental results, the suggested approach, which does not use character boundary markers, greatly enhances performance for multilingual scripts and languages with huge character sets.

Subject Areas

Artificial Intelligence, Information Management

Keywords

Long Short-Term Memory, Optical Character Recognition, Reinforced Learning

1. Introduction

The fundamental task of Optical character recognition (OCR) demands solutions in various fields; thus, numerous methods have emerged to resolve this issue [1]. The present generation of OCR models depends on attention-based sequence-to-sequence (seq2seq) frameworks or Long Short-Term Memory-Connectionist Temporal Classification (LSTM-CTC) systems for processing narrow sub-image sequences into recognition results [2]. These models do without requiring character segmentation while learning how characters relate to one another using transition data [3]. Open-source OCR systems have chosen LSTM-CTC as their core component throughout multiple popular software solutions including Tesseract that produces impressive English character identification results [4].

The accuracy levels achieved by these models are lower for languages that include Chinese and Korean [5]. The large number of available characters within these languages produces a complex recognition challenge for computers because the recognition procedure becomes increasingly difficult [6]. The large number of character classes in Chinese and Korean compared to English creates an extremely difficult recognition challenge because texts must be recognized as individual pairs of characters [6]. The restricted capabilities of present-day LSTM applications in managing complex challenges require a more powerful approach for multilingual OCR under circumstances of large character inventories [7]. This research introduces a multi-block computational architecture that incorporates segmenters along with switchers, together with specialized recognizers to solve the existing problems. This innovative approach enhances OCR precision in documents containing multilingual content, especially when dealing with language sets comprising large and complex characters, including Chinese and Korean patterns.

Chinese and Korean languages need specific solutions for character separation and identification, whereas English has different requirements [8]. The proposed novel multilingual OCR framework optimizes character recognition through dedicated recognizers that operate within a unified system framework. A neural system

comprising three blocks conducts operations starting with the segmenter, followed by the switcher, which leads to multiple recognizers as illustrated in Figure 1. The segmenter first divides a word image into individual character images. Each pixel array of characters in the word image gets transferred to individual recognizers by the switcher module, which recognizes particular characters. Each recognizer proceeds to animate the character sub-image assigned to it. The system manages assignments through its switcher by taking into account the distinctive language elements of spacing and stroke patterns, as shown in Figure 2, despite potential visual similarities between characters. The tool consolidates single-character images into word-level representations so it can execute word recognition despite operating at the character level through its segmenter and switcher. The method supports the scalable recognition of multiple languages through their special character traits by utilizing native language features.

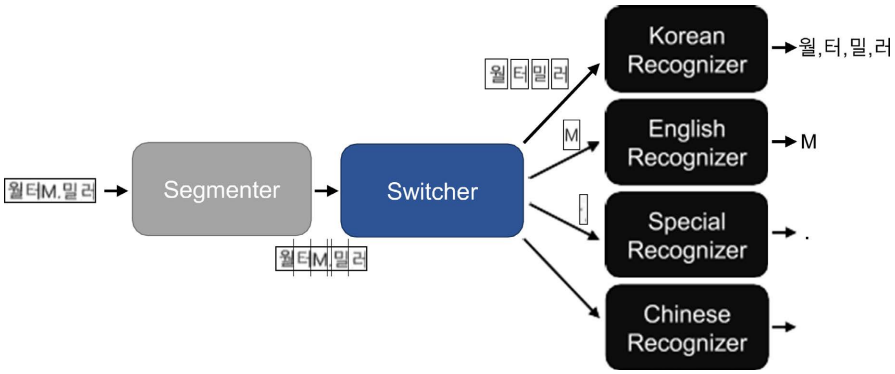

Figure 1. The suggested OCR system's general structure: A word picture is divided into character images by the segmenter, and each character image is given a recogniser by the switcher. The recognisers then carry out the recognition of the allocated.

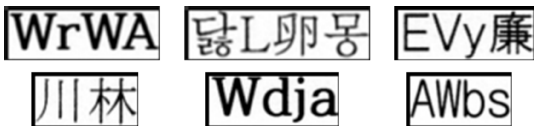

Figure 2. Examples of word images. Character overlap, wide intra-character gaps, and other factors make character segmentation difficult.

The suggested segmenter block's function could appear to be identical to that of traditional character segmentation. In actuality, character-level segmentation served as the foundation for several OCR techniques, and various character segmentation techniques were put forth. For instance, researchers used ad hoc guidelines to determine the character spacing that was the target of projection profiling-based techniques [9]. There are several exceptions, though: As seen in Figure 2, spacing can happen inside individual characters, adjacent characters can touch, and geometric and photometric aberrations can make the spacing between characters unclear. In order to overcome these obstacles, deep neural network-based

Recently, character segmentation techniques were put forth [10]. However, the application of this system to various picture collection contexts is complicated by

the high number of character boundary annotations required by this supervised technique [11]. The segmenter's job is to discover a partition that optimizes the system's overall performance, not only to replicate the ground truth partitions. Notably, as **Figure 2** shows, creating the ground-truth-character bounds is unclear. The number of potential border possibilities increases with the inter-character spacing. Additionally, vertical overlapping may make it impossible to distinguish neighbouring characters. Thus, under the reinforcement learning framework, we train the segmenter block to maximise the edit distance (Levenshtein distance) of the final recognition results [12]. Because it is costly to create a realistic word picture dataset consisting of boundary annotations of characters, this method not only enables us to have a globally optimised OCR system but also minimises annotation efforts [13]. According to experimental results, the suggested OCR system performs very well for a wide range of character-set languages, including Chinese and Korean, as well as their mixed scripts.

2. Related Work

There are essentially two types of recent OCR studies: One is document OCR, which recognises text in document pictures, while the other is scene text recognition, which recognizes text characters in normal scene photographs [14]. Word localisation in document OCR is comparatively simple since backgrounds are often uniform and characters are well-aligned [15]. However, in the test phase, fast throughput is also necessary, and the size of the dictionary in the documented images is considerable. This article discusses two common methods in the field of document picture word recognition, as seen in **Figure 3**.

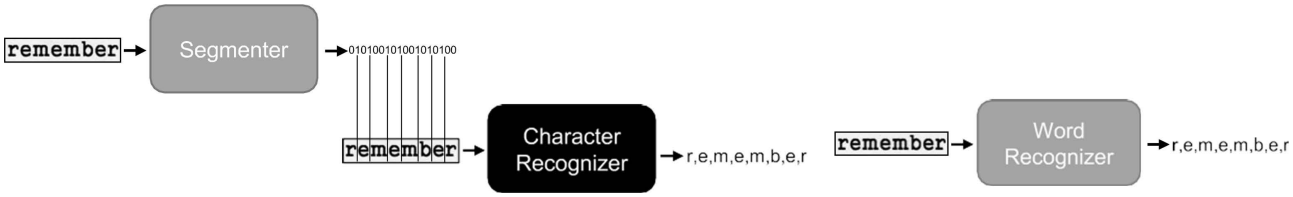

Figure 3. Two common methods for OCR. The OCR methods on the left and right are based on explicit and implicit segmentation, respectively.

2.1. Explicit-Segmentation-Based Approach

A projection-profile-based method, which identifies the borders of characters that have vertical projection profiles, is the simplest way to explicitly segment characters [16]. In this technique, algorithms like morphology or adaptive-threshold approaches were often used to lessen the disparity between projection profile patterns and character boundaries [17]. Nevertheless, projection profile-based techniques experienced touching characters (under-segmentation) as well as over-segmentation, which is spacing inside single characters, despite these ad hoc techniques [18].

Researchers tried using deep learning for character segmentation in an effort to address these issues [19]. For example, [20] employed a Fully Convolutional Network

(FCN) to identify character boundaries after formulating character segmentation as a binary segmentation issue. A lightweight word picture segmentation model was proposed by [21] that chose the most likely boundaries in images using dynamic programming. Character segmentation blocks were trained in a less-than-ideal manner, and the combined optimization of recognition and segmentation has not been addressed.

2.2. Recognition of Characters (With No Explicit Segmentation)

As shown in **Figure 3**, LSTM-based techniques use word pictures as inputs and perform recognition of characters regardless of the need for specific character segmentation. For example, neural networks with three different types of layers, convolution, LSTM, and CTC layers, were employed in LSTM-CTC techniques. The LSTM layers provide the character classes of the input data, while features are extracted by the convolution layers from word picture slices. Since a single character is used to create many slices, the LSTM layers typically generate the same symbol again, which the CTC layer suppresses [22]. Using the encoder-decoder architecture, Seq2seq, utilizing the attention model, was employed [23]. In this case, the encoder forms a vector with one feature that describes the whole input image by scanning word images horizontally. The decoder then uses the attention mechanism to decode the feature vector, producing letter sequences. These techniques may be used in a variety of settings since LSTM-based models do not require character boundary labels for training. Its objective role is also worldwide. Keep in mind that the training of each sub-block within the explicit segmentation of character technique should be done using distinct loss functions. The strategy of employing tiny slices, however, is inappropriate for languages with a high character count. In that instance, due to the two-dimensional juxtaposition of a considerably small number of glyphs, several slices from distinct characters may appear identical, and each slice only offers a restricted perspective of an entire character. As a result, Chinese and Korean characters are less accurate than English, and hybrid scripts are much worse.

3. Proposed Methodology

We can deal with a lot of characters using the explicit segmentation-based technique, and we can employ a global objective function with word-level annotations solely using the implicit segmentation-based approach. As seen in **Figure 1**, we provide a new structure made up of three blocks that combines the best features of both strategies.

This part focuses on presenting the segmenter block after providing a summary of the suggested system.

3.1. System Overview

The objective is to obtain a series of character labels by treating an input word picture as a horizontally concatenated form of images. Initially, a segmenter $S(\cdot)$ receives a text picture (X) as input and outputs the character boundaries.

$$S(X) = (a_1, a_2, \cdots, a_W) \in \{0,1\}^W \quad (1)$$

where, as shown in Figure 3, 1 s represent character boundaries and 0 s represent others. We can divide X into $|a| + 1$ sub-images given a particular map of character boundary $a \in \{0,1\}^W$, where $|a|$ represents the number of 1s inside a. The recogniser accountable for recognising each sub-image is allocated by the switcher $L(\cdot)$. The labels of assigned sub-images are found by recognisers for each language text Ri(.) ∈ {R1(.), R2(.),……..., RM (.)}.

Using a cross-entropy based objective function, L(.) is trained using character pictures from many languages, and then Ri(.) using character pictures from the corresponding language. Next, a segmenter S(.) is trained using the recognisers in reward-evaluation and the trained switcher. The recognition result of X with segmenter S(·) will be Y (X, S(X)), since we indicate the result for recognition of input X employing a map of character's boundary as Y (X, a).

3.2. Modelling of Segmenter

The segmenter receives modeling attention because it divides word image inputs into separate character images. Our method adopts a reinforcement learning approach to address the segmentation task by treating it as a probabilistic system instead of a simple binary classification issue. The central concept models segmentation by describing boundary detection decisions that lead to a sequence determining all character positions in the input image. Specifically, we model the policy distribution π (a|X; θ) that produces actions a ∈ {0, 1} W for a given X for accuracy. To express π (a|X; θ), we use a network's output.

$$f(X;\theta) \quad (2)$$

where θ represents a group of parameters within the scope.

The action, however, is a vector of high dimension and random in nature (in which single arbitrary variables are tightly connected), in contrast to usual reinforcement learning scenarios. For instance, it is extremely rare that $a_i = a_j = 1$ if $|i-j|$ is tiny. This makes the element-wise sampling of action very inefficient; therefore, we add a post-processing step. Due to the inefficiency of element-wise sampling of action, we add a post-pre-processing block to the network output to produce $g(X;\theta) = (p_1, p_2, \cdots, p_W)$ fulfilling:

$$\pi(X:\theta) \cong \prod_{i=1}^{W} P_i^{\alpha} (1-p_i)(1-a_i) \quad (3)$$

where $a = (a_1, a_2, \cdots, a_W)$, θ represents a group of parameters within the scope.

Specifically, we obtain $g(X;\theta)$ by applying three procedures on $f(X;\theta)$. To ensure that the sampling from (3) produces believable examples, we first apply non-max suppression on $f(X;\theta)$ with a window size of 5 ($a_j = a_i = 1$ if $|i-j|$ is tiny will frequently occur). Second, we trim the probability with $\tau = 0.99$ to pose a probabilistic sample, as well as for stronger candidates. Lastly, to exclude low probability candidates early in the training process, we use an element-wise threshold operation:

$$G(X:\theta)=\begin{cases}\min\left(\text{non\_max}\left(f(X:\theta)\right),\tau\right) & f(X:\theta)\geq n\\ 0 & f(X:\theta)<n\end{cases} \tag{4}$$

where n is a threshold value.

While in the training phase, action is drawn from the distribution of policy.

$$a^{(ij)}=\pi(X;\theta) \tag{5}$$

where 'i' and 'j' represent an index of a sample, and assess their incentives according to the edit distance of the final results. In contrast, we choose the most likely course of action during the test phase:

$$S(X)=\arg\_\max_{a}\left[\pi(a\mid X;\theta)\right] \tag{6}$$

for instance, $a_i = 1$ if and only if $p_i \geq 0.5$.

$S(X)$ represents the segmenter function.

3.3. Reinforcement Learning of Segmenter

The reward function is modelled to minimize the edit distance obtained from the segmenter. Instead of creating the entire reinforcement framework, we optimize the reward using the REINFORCE algorithm, much like in [24].

$$\nabla_\theta L=-\nabla_\theta\log\left(f(x:\theta)\right)^{\mathrm{T}}A-\nabla_\theta\log\left(1-f(X:\theta)\right)^{\mathrm{T}}B \tag{7}$$

$$A_i=\sum_j a_i^{(j)}\pi\left(\alpha_i^{(j)}\right)\pi\left(a^{(j)}\mid X,\theta\right)r\left(X,a^{(j)}\right) \tag{8}$$

$$B_i=\sum_j\left(1-\alpha_i^{(j)}\right)\pi\left(\alpha^{(j)}|X,\theta\right)r\left(X,\alpha^{(j)}\right) \tag{9}$$

where T represents transpose, θ represents scope parameters, and x is the input text picture.

A is equal to $(A_1,\ldots, A_W)$ and B is equal to $(B_1,\ldots, B_W)$. Additionally, by lowering the gradient's estimate variances, the author employs the baseline technique to stabilise learning. Equation (6) is utilised as an action baseline as it is unaffected by sampled activities, and the associated reward captures the behaviour of sampled actions, *i.e.*, $b(X) = S(X)$:

$$\dot{A}=\sum\nolimits_j\alpha^j\pi\left(a^j\mid X,\theta\right)\left(r\left(X,\alpha^j\right)-r\left(X,b(X)\right)\right) \tag{10}$$

$$\dot{B}=\sum\left(1-\alpha^j\right)\pi\left(\alpha^j\mid X,\theta\right)\left(r\left(X,\alpha^j\right)-r\left(X,b(X)\right)\right) \tag{11}$$

Lastly, a term for regularizing is added:

$$\nabla_\theta\cong-\nabla_\theta\log\left(f(X:\theta)\right)^{\mathrm{T}}\dot{A}-\nabla_\theta\log\left(1-f(X:\theta)\right)^{\mathrm{T}}\dot{B}-\gamma\nabla_\theta\|\theta\|^2 \tag{12}$$

where γ represents a hyperparameter and θ represents a group of parameters within the scope.

Figure 4 shows the segmenter learning process as a whole.

3.4. Segmenter Training Details

For reliability, the segmenter is first rained utilisng the projection profiles method's

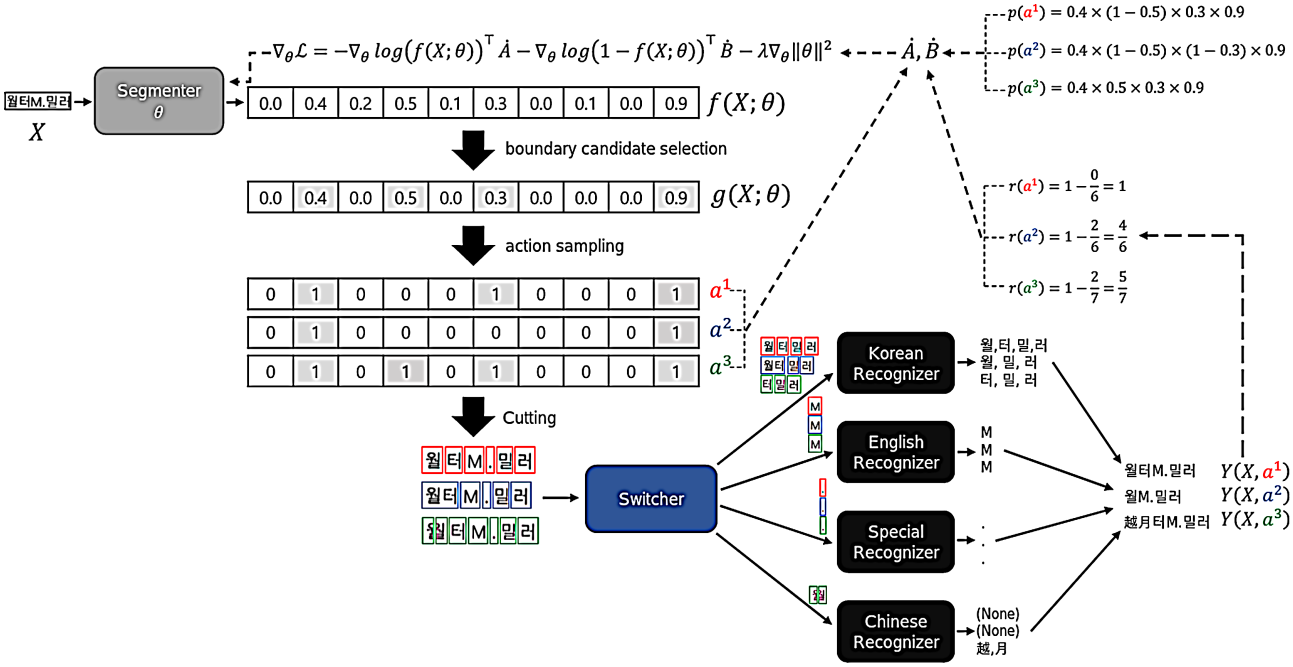

Figure 4. An outline of the suggested approach. The entire stages (reward evaluation, sample-action as well as A, B calculations) are depicted in this figure. Details are in the text.

pseudo labels, implementing a binary cross entropy approach element-wise [25].

$$r(X,\alpha)=1-\frac{d(Y(X,\alpha),Y)}{\max(|\alpha|+1,N_Y)} \tag{13}$$

where N_Y is the number of characters in Y, Y represents the ground-truth text annotation for X, and $d(.)$ is an edit distance.

One may think of the second term in (7) as an edit distance of normalized length. The segmenter's goal function is constructed as follows, depending on the reward: element-wise.

$$\begin{aligned} L(\theta) &= -E_\alpha\left[r(X,\alpha)\right] \\ &\cong -\sum_j \pi\left(a^j \mid X,\theta\right) r\left(X,\alpha^j\right) \end{aligned} \tag{14}$$

where $L(\theta)$ represents the segmenter's goal function.

In order to use the policy gradient approach, we calculate the function (14)'s gradient as: training by giving character boundaries' candidates comparatively high probability, this pre-training offers a decent starting point, even when the segmenter learnt by projection profile approaches performs poorly.

Additionally, we use an incremental learning strategy that gradually lowers η in (4). Learning meaningful rules may take longer when all feasible actions are permitted early in the training process. Instead, we deliver promising candidates early on, then progressively give the ones with lesser promises to border candidates by scheduling η. Up to 20 epochs, we linearly reduce η by 0.015 per epoch after starting it at 0.4.

4. Building the Dataset and Architectures of the Recognizer Networks

Following an explanation of three datasets, each brain block's intricate structure is examined. We concentrate on the combination of 2 Asian tests and English, namely Korean and Chinese, regarded as a particular implementation example, even though there is no limit to the number of possible language combinations.

4.1. Data Set 1: Word Images (Synthetic)

As far as we are aware, no publicly accessible word picture collection with mixed scripts in English, Chinese, and Korean exists. As a result, we create a training dataset of synthetic multilingual word images. In order to do this, we start by compiling terms from Wikipedia to create our dictionaries. Next, we add to the vocabulary by creating words using random combinations of Korean (2341 classes), Chinese (1807 classes), English (52 classes), and special characters (46 classes), including brackets, exclamation points, rests, and full stops. Here, the national language standards and the frequency of their appearance on Wikipedia are used to choose classes for Chinese and Korean texts.

We use these dictionaries to generate artificial word graphics in eight distinct sizes and five different font kinds. We use random rotation in [−2˚, 2˚] and add compression artefacts to synthesized word pictures to make them resemble actual scanned photos. Figure 5 displays a few pictures. In Chinese, there are 208,480 synthetic word pictures, in Korean, there are 314,880, and in English, there are 325,360.

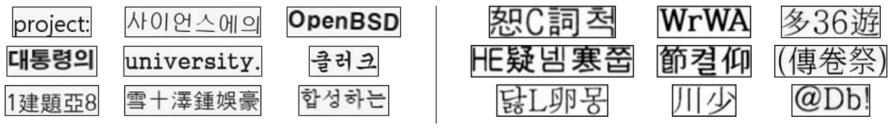

Figure 5. Word picture examples from the dataset. Word pictures on the left were created artificially, and word images on the right were scanned.

4.2. Data Set 2: Test Words

We have used flatbed scanners to scan printed papers for realistic test pictures, and we have applied the following processes to the scanned images: word segmentation, text line extraction, and skew correction [26] [27]. As seen in Figure 5, we concatenated Chinese, English, and Korean text characters to create the test set using multi-script words.

4.3. Data Set 3: Real Character Images

As seen in Figure 6, we gather actual scanned character pictures to train switchers and recognizers. We create them using eight distinct scales and five different font kinds, much like the word dataset. Additionally, we gather character pictures from textbooks and periodicals by cropping characters in scanned photos to cover other font kinds as well as styles of text that were not included in the training dataset.

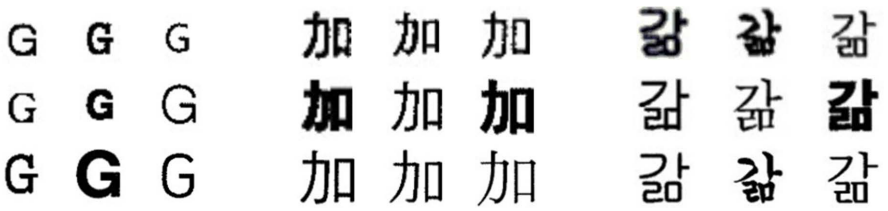

Figure 6. Character picture examples from our dataset. Images of scanned English characters are on the left, Chinese characters are in the middle, and Korean characters are on the right.

4.4. Neural Network Architectures

We employ four recognisers, one segmenter, and one switcher; the specifics of the network architectural setup are covered in this part. Conv (c, $i \times j$, $k \times m$) denotes a convolution layer consisting of channels (c), a ($j \times i$) sized kernel, with vertical strides (k), as well as horizontal strides (m). Table 1 summarizes the architectures of each language recogniser used in this article, while Table 2 describes the architectures of the character language switcher and segmenter. A bidirectional LSTM (layer) of dimension (d) is denoted by the notation BiLSTM (d). Maximum pooling and average pooling layers with ($j \times i$)-sized kernels and $k \times m$ strides are denoted by MaxPool ($j \times i$, $k \times m$) as well as AvgPool ($j \times i$, $k \times m$), respectively. RELU was employed as the activation function to denote the normalization of the batch as well as for convolution layers, Softmax, and Batchnorm.

We designed the segmenter's convolution layers based on the architecture of the character segmenter, which demonstrated high performance with lightweight

Table 1. R_{spe} (Special characters), R_{eng} (English characters), R_{chi} (Chinese characters), and R_{kor} (Korean characters) recognition architectures.

Recognizer	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6	Layer 7	FC Layer 1	FC Layer 2	Output
RENG	Conv (6, 5 × 5, 1 × 1)	AvgPool (2 × 2, 2 × 2)	Conc (16, 5 × 5, 1 × 1)	BatchNorm	Conv (120, 5 × 8, 1 × 1)	AvgPool (2 × 2, 2 × 2)	BatchNorm	FC (54)	FC (53)	Softmax
RSPE	Conv (6, 5 × 5, 1 × 1)	AvgPool (2 × 2, 2 × 2)	Conc (16, 5 × 5, 1 × 1)	BatchNorm	Conv (120, 5 × 8, 1 × 1)	AvgPool (2 × 2, 2 × 2)	BatchNorm	FC (48)	FC (83)	Softmax
RCHI	Conv (12, 5 × 5, 1 × 1)	AvgPool (2 × 2, 2 × 2)	Conc (32, 5 × 5, 1 × 1)	BatchNorm	Conv (240, 5 × 8, 1 × 1)	AvgPool (2 × 2, 2 × 2)	BatchNorm	FC (180)	FC (169)	Softmax
RKOR	Conv (18, 5 × 5, 1 × 1)	AvgPool (2 × 2, 2 × 2)	Conv (48, 5 × 5, 1 × 1)	BatchNorm	Conv (360, 5 × 8, 1 × 1)	AvgPool (2 × 2, 2 × 2)	BatchNorm	FC (84)	FC (29)	

Table 2. Switcher (L) and segmenter (S) architecture.

Module	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	Bi-LSTM 1	Bi-LSTM 2	FC	Activation
S	Conv (6, 5 × 5, 2 × 1)	Conv (6, 5 × 5, 2 × 1)	Conv (6, 7 × 3, 2 × 1)	Conv (6, 5 × 5, 2 × 1)	Conv (6, 3 × 3, 2 × 1)	Bi-LSTM (8)	Bi-LSTM (8)	FC (1)	Sigmoid
L	Conv (6, 5 × 5, 2 × 1)	Conv (6, 5 × 5, 2 × 1)	Conv (6, 7 × 3, 2 × 1)	Conv (6, 5 × 5, 2 × 1)	Conv (6, 3 × 3, 2 × 1)	Bi-LSTM (8)	Bi-LSTM (8)	FC (1)	Sigmoid

designs [28]. But at the conclusion of convolution layers, we change the strides and padding widths such that the resulting feature map is $W \times 1$ from $W \times 32$ inputs. Then, to better use whole word pictures, we apply double layers of bi-directional LSTM on top of them. The segmenter only handles binary classifications, which are column-wise and require a very limited amount of parameters, although it accepts word pictures as inputs.

We use the VGG-13 as a basis model, which has a bigger capacity than LeNet-5, to collect form characteristics in a variety of scripts for the switcher L(.) [29]. However, we decrease the number of channels and layers to increase efficiency.

To take use of each language's unique features, a recognizer is created, with architectures based on language As well as LeNet5 architectures supporting batch normalization and a fully connected layer of English and special character recognizers (Reng, Rspe) which have 53 and 47 classes, respectively along with an extra NR (which means Not Recognizable class), with 1818 classes including the NR class. The R_{chinese} is built to have double the number of channels as the R_{english} in order to recognize Chinese, which has numerous classes and complex patterns.

Figure 7 illustrates the performance of a proposed method in predicting boundaries and OCR results. The upper block displays arbitrary sequences, while the lower block shows real-world mixed cases. Each row presents scanned images, segmenter predictions, OCR outcomes (correct in blue, incorrect in red), and the actual labels for comparison because there are fewer classes and more fundamental components in Korean characters. Specifically, every Korean character may be divided into three fundamental components: the head, middle, right, and the bottom.

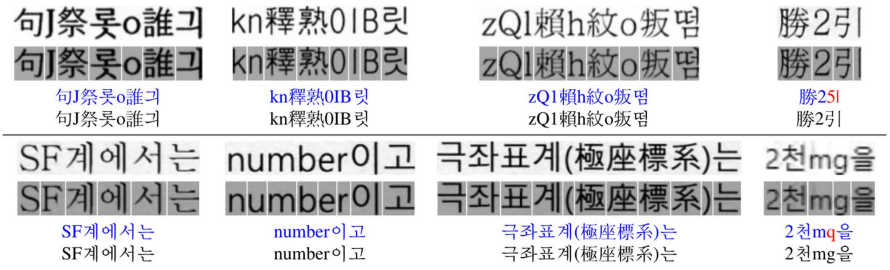

Figure 7. Cases of the suggested method's boundary predictions and OCR outcomes on arbitrary sequences (check upper block) as well as real-world mixed cases (check lower block). The inputs (scanned word pictures) are shown in the first row, followed by the segmenter prediction results in the second row, the OCR results (correct predictions are shown in blue and incorrect predictions in red), and the ground truth labels in the last row.

The final numbers of classes for these components are 20, 22, and 29, respectively, which is significantly less than the size of the entire character set. For these characters, we employ an additional NR class. Therefore, by recognising the labels, we are able to anticipate the characters' labels.

Of all three components simultaneously, without character separation. Be aware that some characters do not require jongseong, and that there are 29 classes total, including:

1. 27 classes of '*jongseong*'
2. 1 case without use of 'jongseong'
3. 1 representing NR.

We employ a three-headed model to identify three items. In other words, a shared block (convolution layers) is used by the Korean recogniser to extract features, and three separate, completely linked blocks then estimate the 3 elements' labels, respectively. For convolution layers, we employ three times as many channels as LeNet-5, and every fully connected layer is the same size as LeNet.

5. Results of the Experiments

We carry out in-depth tests to assess the suggested approach. In our studies, we linearly scale word/character pictures to a height of thirty-two pixels. Since 44 is the maximum character width in the sets, a padding of 0 columns is done to create 44×32 pictures for switcher/recognizers.

The segmenter is trained via reinforcement learning, which requires a trained switcher and recognisers; thus, we train the switcher and recognisers first. Adam optimiser is used to train the switcher and character recognisers over 100 epochs, using a learning rate of 0.0001 and a decay rate of 0.9 every ten epochs. We pretrained the segmenter S(·) using pseudo-labels for 10 epochs with a 0.0001 rate of learning, which is lowered by 10 times every three epochs, as described in Sec. III-D with the conventional character segmentation-based OCR summarised in **Table 3**.

Table 3. Multilingual document accuracy (%). Test-set words with Chinese characters are represented by Chi, English by Eng, and Korean by Kor. Mixed scripts are indicated by the plus symbol "+." "Boundary supervised" indicates that character boundary annotations are necessary for training. Conversely, "word supervised" indicates that word labels are necessary for the training process.

Method	Boundary Supervised	Word Supervised	CHI	ENG	KOR	CHI + ENG	CHI + KOR	ENG + KOR	CHI + ENG + KOR
Tesseract [30]	Yes	No	66	18.12	39.64	30.6	25.64	19.37	5.43
Projection Profile	Yes	No	78.23	24.98	18.95	47.78	76.31	47.59	48.23
Mixed	Yes	No	81.86	93.86	63.93	47.59	48.23	64.85	96.78
Two-Step	Yes	No	64.85	96.78	57.69	52.78	72.28	86.21	50.58
Proposed method	No	Yes	86.21	50.58	77.65	96.2	37.84	66.91	66.46

When it comes to English character recognition, Esteract performs really well. OCR systems based on word-level processing (such as Tesseract utilising the LSTM-CTC model) perform better than character segmentation-based techniques since English has numerous similar characters (such as "l" and "I") that should be detected depending on the context. Tesseract, on the other hand, lacks character-level granularity and performs far worse for mixed-script words than the projection profile-based approach [31]. Furthermore, Tesseract does not work well in character-rich languages such as Chinese and Korean.

Evaluation is also done on the performance of other character OCR techniques,

which are segment-based, such as neural networks (NN) based techniques as well as a projection profile-based technique [32]. Two neural network-based techniques were trained. Given the dataset, and for a fair comparison, the same switcher and recognizer are used. Keeping in mind that the Mixed and the Two-Step approaches need the character borders' ground truth labels, but the reinforcement learning framework eliminates the necessity for these ground truth character boundaries [33]. As demonstrated, they often perform worse than this approach. This result might therefore show the benefits of the global-object function, which directly reflects the quality of the final outcomes, converging to tiny loss values on the artificially created word pictures.

Predicted boundaries are displayed as white lines in **Figure 8**, which displays some character segmentation findings. Our approach can identify the appropriate boundaries of characters considering a variety of scenarios and script kinds, in contrast to existing character segmentation-based techniques.

Additionally, the number of individual modules is analyzed on the failure instances to elucidate the segmenter's role. Focusing on segmentation techniques in the studies using a fixed switcher and recognisers, we assess performance across several languages using Chi + Eng + Kor word pictures.

The segmenter is a crucial part of the performance, as indicated by **Table 4**, which also shows that errors in blocks of segmentation are dominant. Our approach performs better than existing approaches by enhancing segmentation performance.

Figure 8. Comparing different approaches to character segmentation. First row: Enter examples of word images. The results of our technique are shown in the last row, followed by the projected boundaries in the second row, the results in the third row, and the results in the fourth.

Table 4. The number of failures for Chi + Eng + Kor inputs is 1161 characters. We can see that a) given correct character segmentation, other blocks (L, Reng, Rspe, Rchi, Rkor) work well, and b) the segmenter (S) plays a key role in the performance.

Method	S	L	RENG	RSPE	RCHI	RKOR	Total
Projection Profile	606	3	0	0	1	0	610
Mixed	449	19	1	0	2	1	472
Two-step	125	19	1	0	1	1	147
Proposed Method	74	23	1	1	2	3	104

5.1. Ablation Study

Further trials are carried out to determine the effects of training regimens. In order to assess the performance, we first train the suggested segmenter just to replicate projection profile outcomes, that is, without using reinforcement learning. It performs similarly to the projection-profile approach, as shown in Table 5.

After that, the performance was evaluated using the REINFORCE method. We experiment using η equating to 0.1 and 0.4 in this instance, along with the η-scheduling technique that was covered in Sec. III-D. Table 6 indicates that the segmenter's accuracy is significantly increased via reinforcement learning. Additionally, the overall performance is improved by the η-scheduling method, indicating that it aids in the segmenter's convergence. Figure 9 illustrates border prediction examples both with and without reinforcement learning.

Table 5. Ablation analysis of our suggested approach. (percent). We use the same terminology to indicate test set terms as Table 3. "REINFORCE" indicates whether reinforcement learning was used to train the model, and "η" indicates the value of η during the training phase.

REINFORCE	η	CHI	ENG	KOR	ENG + CHI	KOR + CHI	KOR + ENG	KOR + ENG + CHI	Average
Projection Profile	-	76.89	17.43	47.9	57.36	50.95	37.87	46.31	47.46
Proposed Method	0.1	88.93	93.47	66.85	76.95	42.88	94.76	63.54	76.21
Proposed Method	0.3	94.32	97.19	63.54	85.43	86.32	56.3	39.87	86.64
Proposed Method	scheduling	94.76	86.32	85.67	94.2	96.49	80.29	89.34	89.92

Table 6. Architecture of Switcher-Segmenter (SS).

Layer
Conv (32, 5 × 5, 1 × 2)
Conv (64, 5 × 5, 2 × 1)
Conv (64, 3 × 7, 2 × 1)
Conv (128, 5 × 5, 2 × 1)
Conv (128, 3 × 3, 2 × 1)
Bi-LSTM (32)
Bi-LSTM (32)
FC (1)
Sigmoid
Conv (16, 1 × 3, 1 × 1)
Conv (32, 1 × 3, 1 × 1)
Conv (64, 1 × 3, 1 × 1)
Conv (128, 1 × 3, 1 × 1)

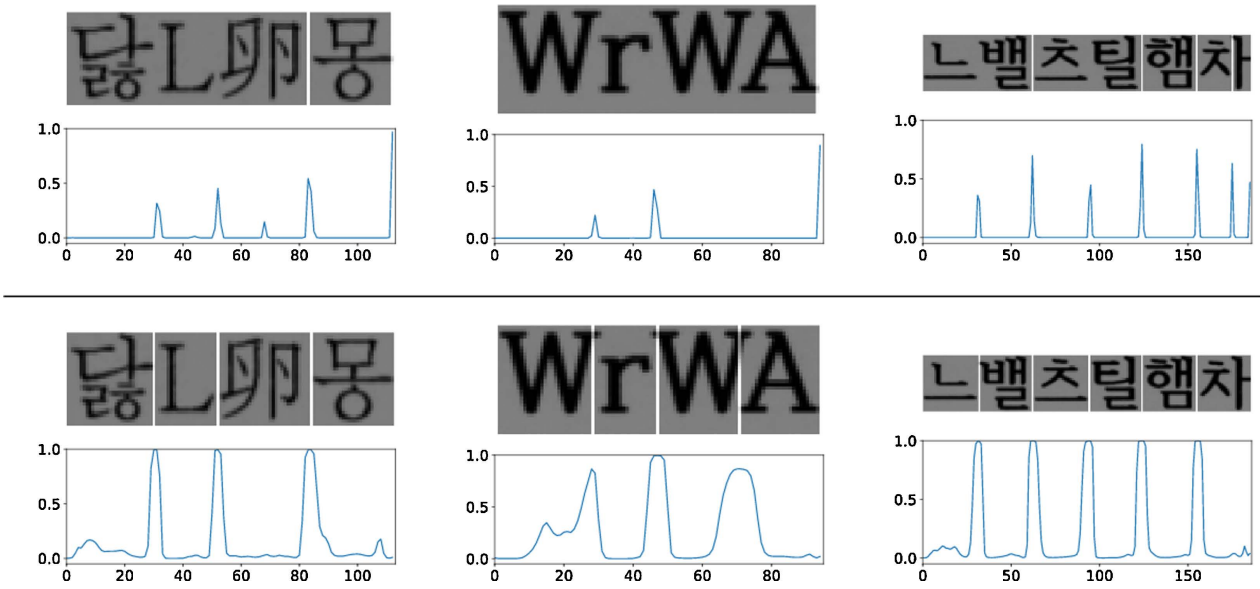

Figure 9. Reinforcement learning effects: results without reinforcement are displayed in the top row, while results with reinforcement learning (containing the η-scheduling scheme) are displayed in the bottom row. $f(X; \theta)$ values are shown by blue graphs. It should be noted that only localized maximum locations of graphs bigger than a threshold of 0.5 are chosen as final character borders since non-max suppression is used to generate character boundaries ($g(X; \theta)$).

5.2. Further Experiments

Further experiments were also carried out with other designs and normalisation techniques in an effort to enhance the effectiveness and performance of the suggested approach.

1) Integration of the Switcher and Segmenter

In order to simultaneously forecast borders and language classes, we attempted to merge the switcher and the segmenter shown in Figure 1 while utilising many independent recognisers. An illustration of our Switcher-Segmenter (SS) designs is shown in Table 6, which is just the combination of the segmenter with the switcher seen in Table 2:

Our SS network is built to produce language predictions at the network's end and boundary predictions within the network's Centre. Increments on the channels in the convolution layers are made, as well as the units in the LSTM layers of the segmenter component, in order to learn the features for both language prediction and border prediction. Additionally, we lower the number of channels for complexity reduction and the height of the kennel of the convolution layers within the switcher component to 1 in order to use consecutive inputs from LSTM layers.

In order to train this two-headed network, one must conduct sequence-based learning as a supplement to the reinforcement learning. The output of the right-hand-side head of our SS network is a sequence whose length is equal to the width of an input image, and the ground truth label is a sequence whose length is equal to the word's character count.

To tackle this issue, we employ a CTC loss that makes sequence-to-sequence learning possible [34]. Similarly, given a H × W input, the SS network produces W

inference results during the inference phase, using just one language class per segmented character image. To solve this issue, we first use the outputs of the left-hand-side head in Table 6 to extract character boundaries. Then, utilise the mean label on the relating interval as the language label for that sub-image. Table 7 displays the outcomes of this SS model. Despite our attempts at training for various configurations, the SS model performs poorly in comparison to the suggested approach. We think that the failure of incentive assessments is the cause of this. Rewards must accurately represent the attributes of actions in order for a model to be trained utilising the reinforcement learning framework. However, promising actions derived from the policy distribution frequently receive poor results because of inaccurate language classes, and the overall model predicts both language classes and border labels.

Table 7. Displays the Switcher-Segmenter model's results.

Method	CHI	ENG	KOR	ENG + CHI	KOR + CHI	KOR + ENG	ENG + CHI + KOR	Mean
Proposed method	93.77	76.1	96.7	86.32	96.84	86.64	89.76	89.79
Switcher_Segmenter	60.83	36.76	83.55	28.59	34.51	32.49	22.92	41.33

Table 8. Architectures of unified recognizer (*Runi*).

Layer
Conv (42, 5 × 5, 1 × 1)
AvgPool (2 × 2, 2 × 2)
Conv (112, 5 × 5, 1 × 1)
BatchNorm
AvgPool (2 × 2, 2 × 2)
Conv (840, 8 × 5, 1 × 1)
BatchNorm
FC (84)
FC (20)
Softmax

2) Integration of the Recognisers

Additionally, we have attempted to create a uniform recogniser that uses a fixed segmenter as well as switcher architecture and shares convolution layers. The network uses 6 heads for the prediction of character classes, as Table 8 illustrates. This universal recogniser is taught to provide the NR class for foreign characters since it creates classes for characters for all languages. We utilise the same fully connected layer topologies in Table 1 and put the channel number in shared convolution layers to the channel's total amount in recognisers for a fair comparison. During the test phase, we select the outputs of this unified recogniser based on the

switcher's result. As seen in Table 9, this combined recogniser works marginally worse despite the extended inference time. As a result, it can be said that single recognisers show more eagerness to utilize the structural features of each language.

3) Techniques of Normalisation

The recognisers were trained using additional normalisation techniques, including layer normalisation and SELU, and convolution layers were used in order to enhance performance [35]. Nonetheless, the outcomes are comparable to the batch normalisation, as seen in Table 10. We think that standard batch normalisation performs well and that CNN training for recognition of single characters does not impose a difficult challenge.

Table 9. Display the outcomes of the unified recogniser models.

Model	CHI	ENG	KOR	ENG + CHI	KOR + CHI	KOR + ENG	ENG + CHI + KOR	MEAN
Proposed Method	93.77	76.87	96.93	86.24	96.9	86.39	89.46	89.79
Recognizer (Unified)	92.48	74.85	94.82	84	95.28	86.25	88.54	88.53

Table 10. Switcher and recogniser results using additional normalisation techniques. The model with layer normalisation is referred to as "LayerNorm". Models having SELU activations are referred to as "SELU".

Model	CHI	ENG	KOR	ENG + CHI	KOR + CHI	KOR + ENG	ENG + KOR + CHI	MEAN
Proposed Method	93.26	76.99	98.12	85.77	96.9	86.64	89.23	89.89
Recognizer: LayerNorm	93.27	72.66	95.56	84.07	95.26	86.64	89.99	89.23
Recognizer: SELU	93.8	72.62	95.67	83.04	95.28	86.77	88.69	87.32
Switcher: LayerNorm	92.17	77.63	89.51	87.55	92.65	84.09	89.73	88.23
Switcher: SELU	94.9	77.92	88.92	86.29	90.35	84.68	86.71	86.58

6. Conclusion

In this paper, we have suggested a multilingual OCR system that combines reinforcement learning for the segmenter with three neural blocks: a segmenter, switcher, and numerous recognisers for various languages. In contrast to traditional approaches for multi-language OCR systems, we used the reinforcement learning method to optimise the edit recognition distance of results, or the overall performance of OCR systems. According to experimental data, our approach works better than traditional approaches. In order to reproduce the results of this experiment, the accompanying code for this paper contains a script titled *environment.yml* as well as a list of requirements, titled *requirements.txt*. The former is responsible for setting up the environment required to run the setup, and the latter is a list of all the libraries used in the setup. The file setup.py in the train and test directories contains a script that automates the whole process of either training or testing according

to the respective directory, and then documents the results.

Conflicts of Interest

The authors declare no conflicts of interest.

References

[1] Memon, J., Sami, M., Khan, R.A. and Uddin, M. (2020) Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR). *IEEE Access*, **8**, 142642-142668. https://doi.org/10.1109/access.2020.3012542

[2] Mathew, M., Mondal, A. and Jawahar, C.V. (2024) Towards Deployable OCR Models for Indic Languages. In: Antonacopoulos, A., Chaudhuri, S., Chellappa, R., Liu, CL., Bhattacharya, S. and Pal, U., Eds., *Lecture Notes in Computer Science*, Springer, 167-182. https://doi.org/10.1007/978-3-031-78495-8_11

[3] Mann, D., Raissi, T., Michel, W., Schlüter, R. and Ney, H. (2023) End-to-End Training of a Neural HMM with Label and Transition Probabilities. 2023 *IEEE Automatic Speech Recognition and Understanding Workshop* (*ASRU*), Taipei, 16-20 December 2023, 1-8. https://doi.org/10.1109/asru57964.2023.10389749

[4] Idrees, S. and Hassani, H. (2021) Exploiting Script Similarities to Compensate for the Large Amount of Data in Training Tesseract LSTM: Towards Kurdish OCR. *Applied Sciences*, **11**, Article 9752. https://doi.org/10.3390/app11209752

[5] Park, J., Lee, E., Kim, Y., Kang, I., Koo, H.I. and Cho, N.I. (2020) Multilingual Optical Character Recognition System Using the Reinforcement Learning of Character Segmenter. *IEEE Access*, **8**, 174437-174448. https://doi.org/10.1109/access.2020.3025769

[6] Chang, C., Arora, A., Garcia Perera, L.P., Etter, D., Povey, D. and Khudanpur, S. (2019) Optical Character Recognition with Chinese and Korean Character Decomposition. 2019 *International Conference on Document Analysis and Recognition Workshops* (*ICDARW*), Sydney, 22-25 September 2019, 134-139. https://doi.org/10.1109/icdarw.2019.40094

[7] Etter, D., Carpenter, C. and King, N. (2023) A Hybrid Model for Multilingual OCR. In: Fink, G.A., Jain, R., Kise, K. and Zanibbi, R., Eds., *Lecture Notes in Computer Science*, Springer, 467-483. https://doi.org/10.1007/978-3-031-41676-7_27

[8] Rho, M., Tian, Y. and Chen, Q. (2024) Word Segmentation for Asian Languages: Chinese, Korean, and Japanese. arXiv:2407.19400.

[9] Yu, Y., Wang, C., Fu, Q., Kou, R., Huang, F., Yang, B., *et al.* (2023) Techniques and Challenges of Image Segmentation: A Review. *Electronics*, **12**, Article 1199. https://doi.org/10.3390/electronics12051199

[10] Ramesh, K.K.D., Kumar, G.K., Swapna, K., Datta, D. and Rajest, S.S. (2021) A Review of Medical Image Segmentation Algorithms. *EAI Endorsed Transactions on Pervasive Health and Technology*, **7**, e6. https://doi.org/10.4108/eai.12-4-2021.169184

[11] Trivedi, A. and Sarvadevabhatla, R.K. (2021) Boundarynet: An Attentive Deep Network with Fast Marching Distance Maps for Semi-Automatic Layout Annotation. In: Lladós, J., Lopresti, D. and Uchida, S., Eds., *Lecture Notes in Computer Science*, Springer International Publishing, 3-18. https://doi.org/10.1007/978-3-030-86549-8_1

[12] Gao, Y., Chen, Y., Wang, J. and Lu, H. (2021) Semi-Supervised Scene Text Recognition. *IEEE Transactions on Image Processing*, **30**, 3005-3016. https://doi.org/10.1109/tip.2021.3051485

[13] Liu, J., Zhong, Q., Yuan, Y., Su, H. and Du, B. (2020) SemiText: Scene Text Detection with Semi-Supervised Learning. *Neurocomputing*, **407**, 343-353. https://doi.org/10.1016/j.neucom.2020.05.059

[14] Gupta, M., Choudhary, A. and Parmar, J. (2021) Analysis of Text Identification Techniques Using Scene Text and Optical Character Recognition. *International Journal of Computer Vision and Image Processing*, **11**, 39-62. https://doi.org/10.4018/ijcvip.2021100104

[15] Neng, H.Z. (2022) Automated Scanned Receipt Processing with Optical Character Recognition and Machine Learning. University of Malaya (Malaysia).

[16] Rexi F, A. and Jacob, L. (2022) Optical Character Recognition System with Projection Profile Based Segmentation and Deep Learning Techniques. 2022 4*th International Conference on Advances in Computing, Communication Control and Networking* (*ICAC3N*), Greater Noida, 16-17 December 2022, 12-16. https://doi.org/10.1109/icac3n56670.2022.10074151

[17] Ptak, R., Żygadło, B. and Unold, O. (2017) Projection-Based Text Line Segmentation with a Variable Threshold. *International Journal of Applied Mathematics and Computer Science*, **27**, 195-206. https://doi.org/10.1515/amcs-2017-0014

[18] Sharma, P. and Sachan, M.K. (2017) A Review on Character Segmentation of Touching and Half Character in Handwritten Hindi Text. *International Journal of Advanced Research in Computer Science*, **8**, 1078-1083.

[19] Minaee, S., Boykov, Y.Y., Porikli, F., Plaza, A.J., Kehtarnavaz, N. and Terzopoulos, D. (2021) Image Segmentation Using Deep Learning: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **44**, 3523-3542. https://doi.org/10.1109/tpami.2021.3059968

[20] Gao, Z., Liu, J., Li, Y., Yang, Y. and He, H. (2020) A Novel Semantic Segmentation Model for Chinese Characters. *IEEE Access*, **8**, 179083-179093. https://doi.org/10.1109/access.2020.3027019

[21] Chernyshova, Y.S., Sheshkus, A.V. and Arlazarov, V.V. (2020) Two-Step CNN Framework for Text Line Recognition in Camera-Captured Images. *IEEE Access*, **8**, 32587-32600. https://doi.org/10.1109/access.2020.2974051

[22] Wick, C. and Puppe, F. (2021) Experiments and Detailed Error-Analysis of Automatic Square Notation Transcription of Medieval Music Manuscripts Using CNN/LSTM-Networks and a Neume Dictionary. *Journal of New Music Research*, **50**, 18-36. https://doi.org/10.1080/09298215.2021.1873393

[23] Feng, L., Zhao, C. and Sun, Y. (2021) Dual Attention-Based Encoder-Decoder: A Customized Sequence-to-Sequence Learning for Soft Sensor Development. *IEEE Transactions on Neural Networks and Learning Systems*, **32**, 3306-3317. https://doi.org/10.1109/tnnls.2020.3015929

[24] Wang, J., Liu, Y. and Li, B. (2020) Reinforcement Learning with Perturbed Rewards. *Proceedings of the AAAI Conference on Artificial Intelligence*, **34**, 6202-6209. https://doi.org/10.1609/aaai.v34i04.6086

[25] Wu, Y., Xing, M., Zhang, Y., Luo, X., Xie, Y. and Qu, Y. (2024) UniDSeg: Unified Cross-Domain 3D Semantic Segmentation via Visual Foundation Models Prior. *Advances in Neural Information Processing Systems*, **37**, 101223-101249.

[26] Liang, Q., Peng, J., Li, Z., Xie, D., Sun, W., Wang, Y., *et al.* (2020) Robust Table Recognition for Printed Document Images. *Mathematical Biosciences and Engineering*, **17**, 3203-3223. https://doi.org/10.3934/mbe.2020182

[27] Kaundilya, C., Chawla, D. and Chopra, Y. (2019) Automated Text Extraction from

Images Using OCR System. 2019 6*th International Conference on Computing for Sustainable Global Development* (*INDIACom*), New Delhi, 13-15 March 2019, 145-150.

[28] Xu, G., Li, J., Gao, G., Lu, H., Yang, J. and Yue, D. (2023) Lightweight Real-Time Semantic Segmentation Network with Efficient Transformer and CNN. *IEEE Transactions on Intelligent Transportation Systems*, **24**, 15897-15906. https://doi.org/10.1109/tits.2023.3248089

[29] Nugraha, G.S., Darmawan, M.I. and Dwiyansaputra, R. (2023) Comparison of CNN'S Architecture Googlenet, Alexnet, VGG-16, Lenet -5, Resnet-50 in Arabic Handwriting Pattern Recognition. *Kinetik*: *Game Technology*, *Information System*, *Computer Network*, *Computing*, *Electronics*, *and Control*, **8**, 545-554. https://doi.org/10.22219/kinetik.v8i2.1667

[30] Patil, S., Varadarajan, V., Mahadevkar, S., Athawade, R., Maheshwari, L., Kumbhare, S., *et al*. (2022) Enhancing Optical Character Recognition on Images with Mixed Text Using Semantic Segmentation. *Journal of Sensor and Actuator Networks*, **11**, Article 63. https://doi.org/10.3390/jsan11040063

[31] Sporici, D., Cușnir, E. and Boiangiu, C. (2020) Improving the Accuracy of Tesseract 4.0 OCR Engine Using Convolution-Based Preprocessing. *Symmetry*, **12**, Article 715. https://doi.org/10.3390/sym12050715

[32] Preethi, P. and Mamatha, H.R. (2023) Region-Based Convolutional Neural Network for Segmenting Text in Epigraphical Images. *Artificial Intelligence and Applications*, **1**, 103-111. https://doi.org/10.47852/bonviewaia2202293

[33] Yi, W., Stavrinides, V., Baum, Z.M.C., Yang, Q., Barratt, D.C., Clarkson, M.J., *et al*. (2023) Boundary-RL: Reinforcement Learning for Weakly-Supervised Prostate Segmentation in TRUS Images. In: Cao, X., Xu, X., Rekik, I., Cui, Z. and Ouyang, X., Eds., *Lecture Notes in Computer Science*, Springer, 277-288. https://doi.org/10.1007/978-3-031-45673-2_28

[34] Tian, J., Yan, B., Yu, J., Weng, C., Yu, D. and Watanabe, S. (2022) Bayes Risk CTC: Controllable CTC Alignment in Sequence-to-Sequence Tasks. arXiv:2210.07499.

[35] Renkin, M. and Rahman, J.S. (2020) Improving the Stability of a Convolutional Neural Network Time-Series Classifier Using Selu and Tanh. In: Yang, H., Pasupa, K., Leung, A.CS., Kwok, J.T., Chan, J.H. and King, I., Eds., *Communications in Computer and Information Science*, Springer International Publishing, 788-795. https://doi.org/10.1007/978-3-030-63823-8_89a