



ResneSt-Transformer: Joint attention segmentation-free for end-to-end handwriting paragraph recognition model

Mohammed Hamdan^{*}, Mohamed Cheriet

Synchromedia Lab, System Engineering, University of Quebec (ETS), 1100 Notre-Dame St W, Montreal, QC H3C 1K3, Canada

ARTICLE INFO

Keywords:

Handwritten text recognition
ResneSt
Transformer
Self attention
Segmentation-free
Lexicon-free
Paragraph transcription
OCR
Encoder-decoder
Image-seq

ABSTRACT

Offline handwritten text recognition (HTR) typically relies on segmented text-line images for training and transcription. However, acquiring line-level position and transcript information can be challenging and time-consuming, while automatic line segmentation algorithms are prone to errors that impede the recognition phase. To address these issues, we introduce a state-of-the-art solution that integrates vision and language models using efficient split and multi-head attention neural networks, referred to as joint attention (ResneSt-Transformer), for end-to-end recognition of handwritten paragraphs. Our proposed novel one-stage, segmentation-free pipeline employs joint attention mechanisms to process paragraph images in an end-to-end trainable manner. This pipeline comprises three modules, with the output of one serving as the input for the next. Initially, a feature extraction module employing a CNN with a split attention mechanism (ResneSt50) is utilized. Subsequently, we develop an encoder module containing four transformer layers to generate robust representations of the entire paragraph image. Lastly, we designed a decoder module with six transformer layers to construct weighted masks. The encoder and decoder modules incorporate a multi-head self-attention mechanism and positional encoding, enabling the model to concentrate on specific feature maps at the current time step. By leveraging joint attention and a segmentation-free approach, our neural network calculates split attention weights on the visual representation, facilitating implicit line segmentation. This strategy signifies a substantial advancement toward achieving end-to-end transcription of entire paragraphs. Experiments conducted on paragraph-level benchmark datasets, including RIMES, IAM, and READ 2016 test datasets, demonstrate competitive results compared to recent paragraph-level models while maintaining reduced complexity. The code and pre-trained models are available on our GitHub repository here: [HTTPSlink](https://github.com/mohammedhamdan/ResneSt-Transformer).

1. Introduction

Historical documents (HD) often require optical character recognition (OCR) systems to extract text data. One popular approach is the handwritten text recognition (HTR) model. HTR can identify individual characters within an image of handwritten text [1,2]. Conventional methods for achieving this involve the segmentation and transcription phases. However, unlike printed texts, handwriting images are challenging to split into characters due to various reasons such as: cursive text, inter-intra class variations and quality of background images. Early algorithms attempted to use heuristic over-segmentation to compute segmentation hypotheses for characters, which were then scored as a whole. Over the decades, significant research has been dedicated to studying word-level and line-level segmentation-based techniques for handwriting.

Despite significant advancements, the HTR problem remains challenging due to the inherent variability in handwriting between and

within classes. Segmentation-free approaches have been developed to overcome this, which take a whole word image as input and generate a series of scores. These scores can then be decoded using a lexicon to retrieve the original sequence of characters. However, when dealing with paragraph image recognition, researchers face additional challenges. For example, the modeling framework must include mechanisms for detecting and recognizing text regions within paragraph images. Additionally, the model must establish a reading order to retrieve the final paragraph transcription from all the seen text parts. Handwritten paragraphs exhibit various patterns, such as horizontal and vertical alignment, skew, and slanted text, further complicating recognition.

The accuracy of the preceding text detection and segmentation steps often limits a brief background about HWR models. Motivated by this, Wigington et al. [3] presented a deep learning model that jointly learns text detection, segmentation, and recognition mainly using images without detection or segmentation annotations. The study

^{*} Corresponding author.

E-mail addresses: mohammed.hamdan.1@ens.etsmtl.ca (M. Hamdan), mohamed.cheriet@etsmtl.ca (M. Cheriet).

by [4] aims to accomplish two main objectives. First, it endeavors to create two distinct datasets: the Mayek27, which consists of 4900 isolated characters from the Meitei Mayek alphabet, and the MM (Meitei Mayek) dataset, which contains 189 pages of comprehensive handwritten text. For the lack of training data [5] proposed a random text line erasure approach that randomly erases text lines and distorts documents. Yousef et al. [6] present a newly developed neural network module, OrigamiNet. This module can enhance any fully convolutional, CTC-trained single-line text recognizer into a multi-line version. This is accomplished by providing the model with the adequate spatial capacity to effectively collapse a 2D input signal into a 1D representation while maintaining information integrity. The proposed method is deemed novel and straightforward in design. To this end, Dolfing [7] investigated an end-to-end inference approach without text localization which takes a handwritten page and transcribes its full text. Sharma et al. [8] presented a fully convolution-based deep network architecture for cursive handwriting recognition from line-level images. Singh et al. [9] presented a Neural Network-based Handwritten Text Recognition (HTR) model architecture that can be trained to recognize the entire handwritten or printed text page without image segmentation. Authors in this study [10] presented an efficient procedure using two state-of-the-art approaches from the literature of handwritten text recognition as Vertical Attention Network and Word Beam Search. It is stated that the approach being discussed requires additional physical segmentation annotations to train the segmentation stage. To address this issue, Coquenot et al. [11] proposed an end-to-end approach that performs handwritten text recognition for the entire document, thereby eliminating the need for additional physical segmentation annotations.

Text extraction can be challenging because various factors can pose difficulties in accurately extracting text from images or other media. However, Nag et al. [12] present a novel unified method for tackling these challenges. According to the authors, [13], their model contains an innovative process of mapping an image to a sequence of characters corresponding to the image's text by combining deep convolutional networks with recurrent encoder-decoders. In their work, Carbonell and colleagues [14] introduced an integrated model designed to concurrently carry out the tasks of detecting, transcribing, and recognizing named entities in the handwritten text at the page level. This comprehensive approach allowed the model to take advantage of common features across these tasks, enhancing its overall performance. In this study [15], the authors presented an approach for determining whether a document scan contains handwriting. Authors [16] presented a line segmentation algorithm for Urdu handwritten and printed text and subsequently to ligatures of binding two characters or words together. Kaur et al. [17] proposed a holistic approach and eXtreme Gradient Boosting (XGBoost) technique to recognize offline handwritten Gurumukhi words. This raises the question: are segmentation-free strategies the best solution to HTR? Peng et al. [18] proposed a method that utilizes a simple yet efficient fully convolutional network for recognizing handwritten Chinese text.

Advancements in text extraction from images are evident across various approaches [19–22]. These include deep learning models for license plate recognition, semantic graph embedding techniques, algorithms for sorting characters in multi-line license plates, and tools combining computer vision and machine learning for efficient table textual extraction. All these strategies signify substantial progress in text extraction and recognition technology.

Most existing methods for handwriting recognition use two stages: text segmentation and text recognition. The first stage involves segmenting the text using a hidden Markov model (HMM) [23,24], while the second stage was focused on text recognition. Although each stage could produce good results independently, they have some fundamental flaws. For instance, manually constructing ground truth segmentation and transcription labels at the line level is costly and time-consuming. Moreover, any segmentation mistakes can lead to recognition errors,

which can impact the system's overall accuracy. Additionally, modifying one stage may require retraining the other stage to ensure optimal performance. Furthermore, explicitly segmenting the text presents the issue of how to define a line, which can be a challenging task.

Our proposed approach addresses traditional segmentation-based models' limitations by introducing an end-to-end HTR model that does not require prior segmentation or a constrained lexicon. We synthesized and augmented existing paragraph-level datasets to achieve this and developed an optimized pipeline for our HTR model. The attention mechanism in each block of our proposed architecture plays a crucial role in helping the model learn robust line representations within a paragraph and decode the corresponding language dependencies of character sequences. We use a Convolutional Neural Network (CNN) split attention (ResNeSt50) for feature extraction, generating a two-dimensional feature map for each image by combining convolution with depth-wise separable convolutional modules. An encoder-decoder transformer-based multi-headed self-attention is the transcriber method, making our model capable of retrieving textual information from an input image, regardless of document restrictions such as text size, style, or layout. Our experiments have shown that our proposed model performs better than traditional segmentation-based models, slightly improving HTR performance. As a result, laborious, error-prone ground truth segmentation and transcription labels are no longer needed at the line level. In summary, the following contributions are made:

- Our research presents an innovative approach for HTR that involves a one-stage segmentation-free pipeline with joint attention mechanisms, enabling end-to-end transcription of the entire paragraph.
- In this study, the focus is on exploring the combination of SOTA deep learning methods for HTR to alleviate the challenges posed by complex segmentation hypotheses. The approach is to shift from line segmentation to processing the entire paragraph, which has consistently improved HTR performance by simplifying the implementation process.
- To achieve this, we leverage the latest state-of-the-art vision and language models, such as split and multi-head attention neural networks (ResNeSt-Transformer). These models allow us to bypass the need for line segmentation or a pre-defined lexicon, making the transcription process more efficient and accurate.
- Data augmentation was performed on synthetic paragraph images using the IAM line dataset, READ2016 line dataset, and the RIMES line dataset, which contributed to the competitive performance of the model using only a few publicly available paragraph annotations.
- The proposed architecture is tested on three benchmark datasets, including IAM, READ2016, and RIMES, and produces results that are competitive with those achieved by traditional methods. However, our less complex approach represents a significant step toward a more streamlined transcription process.
- The proposed model based on segmentation-free and lexicon-free techniques demonstrates excellent generalization power, as evidenced by both quantitative and qualitative results.

The study conducted experiments on three public datasets of handwritten paragraphs, Rimes, READ 2016, and IAM. It achieved competitive results with state-of-the-art models that use ground-truth paragraph segmentation. As for the rest of the paper, Section 2 discusses related methods and modeling choices. Section 3 describes the proposed modification and provides system details. Section 4 presents the experiment results. Section 6 presents a brief discussion that compares the proposed system with other methods, suggests potential improvements, and discusses the challenge of applying it to full documents.

2. Related works

The literature review highlights that previous research on text recognition has mainly focused on recognizing single lines of text, with relatively little attention given to identifying entire paragraphs. The study categorizes the methods used in this research according to two key characteristics: (a) segmentation-free methods, which do not require explicit text line segmentation before recognition; and (b) segmentation-based approaches, which typically involve explicit text line segmentation before recognition.

2.1. Segmentation-free text recognition methods

Multiline recognition systems, as described in several studies [25–29], recognize entire paragraphs or pages without initial segmentation. These systems use CNN-attention coupled with decoder attention networks, such as LSTM attention with the CTC function. Some studies, like [25,26], use attention mechanisms for paragraph recognition tasks with implicit line and character segmentation. The authors [30] proposed attention blocks with encoder–decoder architectures at the line and character levels. The subnetwork decoder (LSTM) generates output based on the characters’ likelihood computed from the representation. These topologies require line-level image pre-training but do not require line breaks at the transcription label. Another study by [31] proposed an implicit segmentation Urdu character recognition system in which an MD-LSTM-based recognition engine is trained on statistical features.

Bluche et al. [26] proposed a model that can identify a single paragraph without making assumptions about the document’s layout or size. However, this technique required enormous memory requirements and lacked GPU acceleration for MDLSTM training. Additionally, the inference time was unacceptable, eventually abandoning this technique. Later, an extended work by Bluche et al. [25,32] proposed a new model that could recognize paragraphs by applying the CTC encoder approach and the Multidimensional (MDLSTM) attention model, which outputs a single text-line via an automatic line segmentation method. This approach is much better than the single-line recognition model. However, it has a hard-coded assumption that the text line extends horizontally from left to right (LR - Latin scripts) and fills the entire input image. As a result, this approach only works on paragraph segmentation and fails to handle text layouts. Furthermore, this approach cannot output a variable-length sequence, and a hard separator joins the predicted lines. Therefore, there is a lower likelihood that the model will accurately recreate blank lines and indentations.

Kaltenmeier et al. [33] proposed a segmentation-free Hidden Markov Model (HMM) based method that can output the sequence scores of the image text (word level) with the help of the lexicon dictionary used for the decoding step. Unlike Bluche et al.’s approach, this method does not rely on segmenting text lines or paragraphs. Instead, it recognizes individual words based on their probabilities and the lexicon dictionary. This approach is advantageous when dealing with images that contain irregular or distorted text. However, the model’s accuracy depends heavily on the quality of the lexicon dictionary. Therefore, building a high-quality lexicon dictionary is crucial to achieving the best results.

Coquenot et al. [34] introduced the vertical attention network (VAN) - a cutting-edge solution for recognizing paragraph-level documents using a hybrid attention mechanism to process paragraph images line by line iteratively, relying on line annotations. This unique approach enables VAN to implicitly segment the text while accurately recognizing character sequences associated with each line. With this innovative technique, VAN further elevates the accuracy and efficiency of its text recognition capabilities.

2.2. Segmentation-based recognition methods

HTR models aim to recognize text strings in scanned documents, including Historical Documents (HD). Sueiras et al. [35] proposed a sequence-to-sequence deep neural network model that uses a horizontally sliding window over word image patches. Sueiras’s model performed best on the word level of the IAM and RIMES benchmark datasets using the MDLSTM/CTC architecture. Similarly, the work of Ma et al. [36] introduced a novel multi-scale attention network to extract a robust representation of text images and improve performance by replacing the fully connected layer with a global maximum pooling layer. In contrast to printed text, handwritten scanned documents are challenging to segment into different characters, and the traditional approach [37–39] involves an initial segmentation step followed by transcription. Traditional machine learning methods with hidden Markov models (HMMs) were employed in the early stages of developing HTR systems. When character segmentation was required for further text processing, the work of Mohamed et al. [40] and Mou-Yen et al. [41] attempted to calculate character segmentation hypotheses by performing heuristic over-segmentation and evaluating groups of segments.

Unlike [25,26], [3] do not require a pre-segmentation as earlier approaches, such as those developed by [3,42]; they focus on paragraph and full-page handwritten text recognition. The latter techniques [3] on training data require a prior selection for text lines (ground-truth) to learn text line localization by individual networks or subnetworks in a multitasking network. Line breaks are marked on any textual ground-truth transcriptions. The idea of adapting [3] as a combinatorial optimization problem presented by [42] in a weakly guided fashion in which the authors suggested aligning the anticipated transcription and the corresponding ground truth. These methods established the alignments to be greedily resolved without the requirement for transcription line breaks. However, [42] takes the same amount of pre-training as [3] and performs at a lower level than either of the previous examples.

Bluche and Messina [26], and Puigcerver [29] followed the trend toward a more parallel architecture by replacing the MDLSTM encoder with CNN while still relying on CTC. The segmentation and recognition text line tasks are presented as two distinct networks by [43] inspired by object recognition methods. By focusing on modular techniques, the authors proposed a pipeline and described it in detail in [43]. The passages distinguish handwritten text areas, apply object recognition-based word-level segmentation is made, and the words are arranged logically. In contrast to [43], which focuses on line-level recognition, the authors of [44], taking an approach similar to that of [43], offers an end-to-end paradigm based on word-level recognition rather than line-level recognition.

MDLSTM [45], hybrid CNN coupled with Bidirectional (BLSTM) [46] attention encoder–decoder [47] and Gated fully convolutional network (GFCN) [48] are examples of improved techniques. CNN coupled with MDLSTM [49] can identify the beginning of each line reference, and MDLSTM can recognize text lines by using a special line ending token. Particularly suitable in multiline and full paragraph texts where CNNs operated as segmented multiline predictors, as in [3,42]. The iterative procedure then builds a normalized line by predicting the next location depending on the present position until the end of the line. Finally, CNN + BLSTM served as OCR for line-level recognition. The network will only operate with transcription labels in this situation. Unlike the work of [3,42] that is based on a segmentation-free technique to handle transcriptions without line breaks at the paragraph level.

2.3. Deficiencies in current models

Though [25] is comparably faster to train than [26], in both works, their encoder subnetworks require pre-trained on isolated-line images before training on paragraph images. Hence, [25,26] are slow and hard to train, unlike our experiments. Our proposed model is straightforward

to train directly on paragraph images. Whereas most existing HTR models need prior image segmentation [50] to extract text components such as characters, words, and lines, these approaches have several flaws. These methods of image-text segmentation rely on heuristics or feature engineering that break under severe data changes. Many segmentation-based methods help correct slanted text, curvature, and bold, using wrong properties and heuristics. More critically, it is challenging to separate text units cleanly in real-world handwriting. Lines, for example, may be warped or merged with non-textual symbols and other visual elements. The literature has further information on these limitations [25,26]. Synthesizing a general transcription from external transcribed text segments considers a different technique that might introduce errors and decrease performance.

Several methodologies have been investigated to address the challenge of extracting text from various image sources. Onim et al. [19] introduced BLPnet, an end-to-end DNN model designed explicitly for Automatic License Plate Recognition (ALPR) systems focusing on Bengali characters. This model surpasses existing YOLO-based ALPR models regarding number-plate detection accuracy and outperforms the Tesseract model regarding time requirements. Etaiwi [20] proposed a novel approach named SemanticGraph2Vec for semantic graph embedding. This method considers the semantic relationships between vertices during the learning process, enhancing the overall performance of graph embedding techniques. Minhuz et al. [21] presented an algorithm for sequential sorting of discrete and connected characters found in multi-line license plates. By employing image processing techniques, their algorithm contributes to developing an automatic license plate recognition (ALPR) system capable of accurately distinguishing individual characters even from license plates of inferior quality. Colter et al. [22] emphasized the significance of data extraction from tables and highlighted the necessity of automated table extraction for practical data mining. The proposed tool, Tablext, combines computer vision and machine learning methods to identify and extract textual data from tables efficiently, streamlining the data-mining process.

Additionally, the Right-to-Left (RL) direction scripts like Arabic need stitching because text portions are typically concatenated with a space or new line. Thus, the formatting and indentation may be lost. This limitation indicates that the smaller text size may overlap and intersperse with the more significant parts of information that have not been transcribed. Luckily, the proposed model avoids the issues by implicitly processing and learning from data end-to-end. So, our proposed model is easy to implement with the best choice of hyperparameters. They can jointly perform segmentation and recognition tasks using cutting-edge methods of training and searching. Without human intervention, the training is carried out automatically on paragraph images.

3. Methodology

In this section, we define the problem of paragraph handwriting recognition. Then, we provide an overview of the proposed model and the relevant details about its components. We continue with a detailed description of the Seq2Seq architecture used by our model. This description explains the details of the three module components: split attention convolutional feature extraction, encoder transformer layers, and decoder transformer layers, respectively. Fig. 1 depicts the main components of the complete model architecture.

3.1. Defining the problem

Since we address the paragraph level of the HTR framework, there is no need to predefine image regions as text or non-text. The paragraph image corresponds to its textual ground truth represented as X , and Y denotes the paragraph handwritten images X, Y . The vocabulary contains different characters, including capital (A–Z) and small (a–z) characters, digits (0–9), and the most common spatial characters and

symbols, including the white space. The proposed model can interpret the visual information and the text of the corresponding transcription from the paragraph image and the paragraph text, where ($x_i \in X$) and their associated strings ($y_i \in Y$). We use the benchmark datasets in academia, the IAM benchmark dataset [51] and RIMES [52], which have more than 600 writers in English and French scripts. Therefore, it persists in challenging handwriting, inter-class, and intra-class variability.

Preferably, a visual feature representation encoder targets extracting valuable information using ResNeSt50 and Encoder Transformer from the paragraph images. We use the attention mechanism and positional encoding techniques to learn the paragraph-handwritten image characteristics and direct its attention to various characters' positions. Following that, the text transcriber is dedicated to producing the decoded characters and simultaneously attending to visual and corresponding text transcription. The proposed model is an end-to-end trainable model that learns handwritten visual image representations and their textual ground truth.

Fig. 1 illustrates the architecture of the proposed approach where ResNeSt50 [53] and encoder-decoder transformer [54] make up the bulk of the proposed model.

3.2. Model overview

HTR models often struggle with obtaining accurate line segmentation, complicating the training and transcription process. To address this, a state-of-the-art solution using ResNeSt-Transformer models combines vision and language models through joint attention, providing an end-to-end recognition of handwritten paragraphs. This one-stage, segmentation-free pipeline, consists of three modules: feature extraction, encoding, and decoding, each utilizing attention mechanisms and positional encoding. By employing joint attention, the neural network implicitly segments lines, enabling end-to-end transcription of paragraphs. Tested on RIMES, IAM, and READ 2016 datasets, this approach achieves competitive results with reduced complexity, paving the way for fully automated handwriting transcription. In Fig. 1, the ResNeSt-Transformer model employs a one-stage segmentation-free pipeline that uses joint attention mechanisms to handle a paragraph image in an end-to-end trainable fashion. The pipeline consists of three modules, each building upon the output of the previous module. The first feature extraction module utilizes CNN with a split attention mechanism (ResNeSt50). Then, an encoder module with four transformer layers generates robust representations of the entire paragraph image. Finally, a decoder module with six transformer layers creates weighted masks. A Split Attention Network is used for handwriting paragraph recognition by splitting the input image into smaller parts and processing each part with its attention mechanism.

The encoder and decoder modules are coupled with a multi-head self-attention mechanism and positional encoding. This enables the model to focus on the current time step on specific feature maps. By leveraging joint attention and a segmentation-free model, the neural network computes split attention weights on the visual representation, allowing for implicit line segmentation. Images are processed using a CNN for features extractor, then the Transformer encodes and decodes these features using multi-headed self-attention layers. A joint attention module decodes the paragraph image at the character level. The proposed method uses the backbone ResNeSt50 as a convolutional network to extract the 2D features representation. Before feeding data through a transformer encoder, we add the 2D positional encodings to the feature vector. We applied four encoders and six decoder layers; each has self-attention and Feed Forward Neural Network (FFNN).

Fig. 2 illustrates the flowchart of the proposed model. The process begins with the input image and target text as the initial inputs. The image undergoes feature extraction using a ResNeSt-50 backbone model, which extracts meaningful features. These features are passed through a 2D convolutional layer, converting them into 256 feature planes. 2D

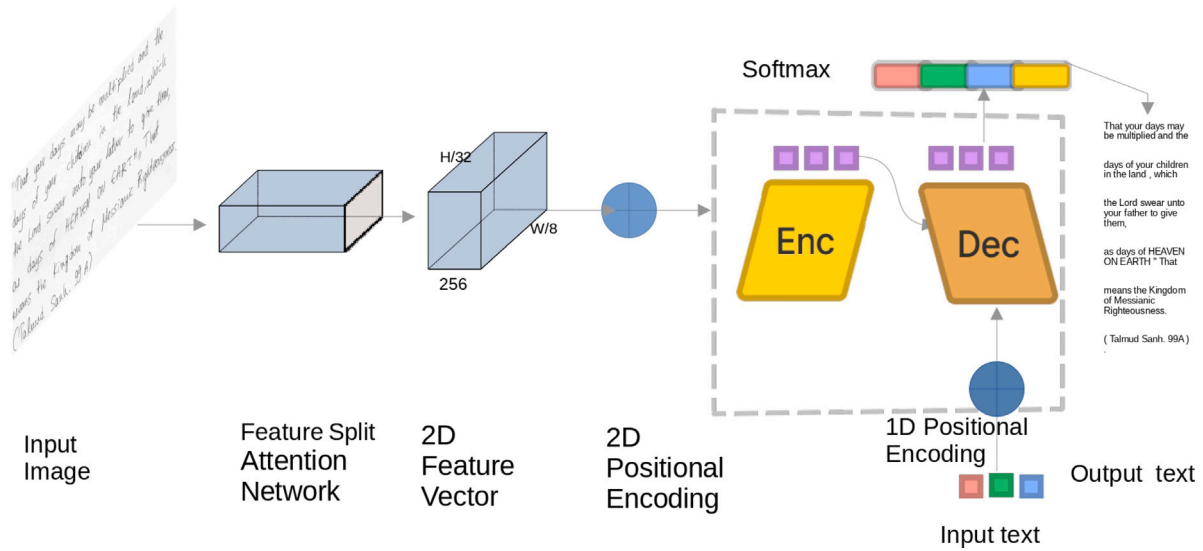


Fig. 1. Network architecture overview: (left) A paragraph input image is fed to a CNN split attention that serves as the backbone for the feature extraction method. (right) A 2D positional encoded technique encodes the CNN output feature vector before feeding it to the encoder self-attention transformer layers. The CNN output feature representations serve as input to the decoder self-attention layers. The embedded ground truth is encoded with the same 1D position encoded technique before feeding it to the decoder self-attention layers. Finally, a Softmax activation function outputs the probability for each class.

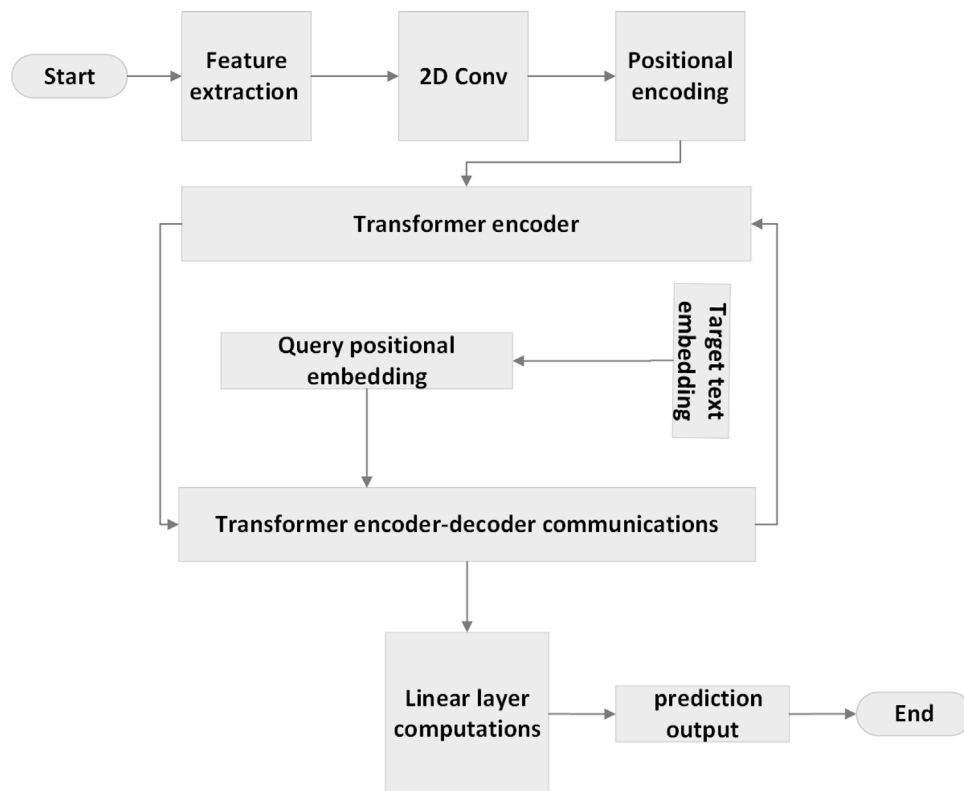


Fig. 2. This flowchart illustrates the sequential steps involved in the HPR process.

positional encodings are generated and concatenated with the feature maps to capture spatial information. The concatenated feature maps and positional encodings are input to the transformer, comprising encoder and decoder components. The transformer utilizes multi-head self-attention mechanisms in the encoder and decoder blocks to capture dependencies between features and positional encodings. The target text is embedded and processed through a 1D positional encoding layer. The decoder applies multi-head self-attention to the feature maps and positional encodings. The final output is the predicted text.

This flowchart demonstrates the sequential steps in feature extraction, encoding, decoding, and prediction, highlighting the model's ability to recognize handwritten text.

3.3. Feature visual representation

Encoding visual feature representation using ResNeSt [53] similar to ResNeXt [55], which applies a multi-path feature extraction of the input image. Then in each path, divided attention is employed. Similar

to SKNet [56], the split is accomplished by performing convolution with various kernel sizes to select the optimal kernel size channel-wise for each. ResNeSt, conversely, does convolution with kernels of the same size as a branch selection. Several models are generated and then assembled using divided attention by ResNeSt.

High-level feature representations are extracted from the handwritten paragraph image ($x_i \in X$) in this feature visual encoder stage, where x_i represents a sample of input images as in Fig. 1. To process the handwritten paragraph images, the input image x_i is first processed by the CNN, which can handle images of any size. An intermediate F_v representation of visual features is constructed of size ($f = 2048$) as the feature vector. The ResNeSt50 convolutional architecture serves as the backbone of the HTR architecture. Compactable visual feature representations that give a contextualized global view of the whole input image are rare. Therefore, the attention mechanism layer in ResNeSt50 helps extract this meaningful information.

3.4. Positional encoding

Text images in Latin scripts are typically processed sequentially, moving from left to right. In this process, the positional encoding stages are strategically positioned before the transformer encoder phase. These stages are designed to capture and encode this essential sequence information without unnecessary repetition. Such encoding assists the model in determining the precise location of the next character in the text paragraph. To utilize order sequences effectively, the model leverages positional encoding (PE) as formulated in Eqs. (1), (2). This approach aids in encoding the tokens' positions within the sequence [54]. Consequently, we integrate the input embeddings with the PE at the decoder networks, visually represented in Fig. 1. Furthermore, we employ a 2D learnable PE [57] on the feature vector that the ResNeSt encodes after adjusting the vector to the same hidden dimension (256) matching the expected input of the transformer model. The inclusion of PE in our proposed architecture is highly beneficial. It instructs our model to focus actively on the specific position within the sequence, thereby enhancing its performance.

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right) \quad (1)$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{\frac{2i+1}{d_{model}}}}\right) \quad (2)$$

where the equations denote that the character at a particular position i is represented by pos , while the hidden dimension size in the transformer block is denoted by d_{model} . The Eqs. (1) and (2) include absolute positional information for the encoder at the decoder input.

3.5. Self-attention module

The self-attention layer is used for the image feature vector to improve the visual representation further. Inspired by Vaswani et al. [54], we used 4 heads of a multi-head self-attention module. The module is described as taking in three inputs that correspond to query, key, and value, which are represented by \hat{F}_v as Q_v , K_v , and V_v , respectively. The correlation information of the attention visual feature is represented as A_v^i and is obtained through the use of Eq. (3),

$$A_v^i = softmax\left(\frac{q_v^i \cdot K_v}{\sqrt{f}}\right)V_v \quad (3)$$

where the $q_v^i \in Q_v$ as input query and i range from (0 and $w-1$), K_v and V_v are the input key and value, respectively. Finally, from Eq. (3) where $\hat{F}_v = \{A_v^0, A_v^1, A_v^2, \dots, A_v^{w-1}\}$, we obtain the high level visual representation.

3.6. Textual representation and transcription

The text encoder and decoder are two model modules depicted in Fig. 1. Its goal is to output decoded characters using visuals and the knowledge of language-specific to create textual representations. Encoder-decoder transformer-based systems to learn n-grams from the textual transcriptions, predicting the next most likely character. Using Eq. (4) Text encoding, Eq. (3) for language self-attention, and Eq. (5) mutual attention in order to make up the text transcriber as the final output in Eq. (6). To properly process the paragraph-level string, we need a small number of symbols with no textual content in addition to the large vocabulary size of the dataset alphabet. $\langle SOP \rangle$ is used to indicate the beginning of a new paragraph, $\langle EOP \rangle$ the end of the paragraph, and $\langle P \rangle$ to indicate padding for the short length of a given paragraph. In the prediction, the transcriptions $y_t \in Y_t$ are lengthened to a maximum number of characters C for a given paragraph.

Utilizing the same positional encoding Eqs. (1), (2) in Eq. (4), the model embeds the characters with the help of the final condense connected layer that transfers every symbol or character in the input string to its corresponding vector representation with a dimension of 256. Since each decoded character is sent back into the decoder to help anticipate the next character, the sequence-to-sequence techniques [58,59] prevent parallelization from occurring in the decoding phase. Thankfully, We leverage the transformer strategy [54], where all decoding stages are provided simultaneously with a masked procedure.

$$F_t = Mapping(y_t) + PE \quad (4)$$

where the output shape of the textual feature representation (F_t is (f,C)). In Fig. 3, we randomly picked different positions of heat maps; due to the limited space, we could not visualize the whole paragraph heat map at the decoder layer Multi-head self-attention.

When the self-attention module implicitly produces n-gram-like features, forming the textual feature representation \hat{F}_t , which seeks to condense the text information further and acquire language-specific attributes. Concerning visual self-attention, as defined in Eq. (3), we refer to the textual feature representation as \hat{F}_{vt} . This representation is equivalent to Q_{vt} , derived from the textual features denoted as \hat{F}_t . For our model, we define the query, key, and value as Q_{vt} , K_v , and V_v , respectively. We denote the correlation information derived from the attention textual feature, calculated by Eq. (5), by A_t^i ,

$$A_t^i = softmax\left(\frac{q_t^i \cdot K_v}{\sqrt{f}}\right)V_v \quad (5)$$

where the $q_t^i \in Q_t$ as associated ground truth input query and i range from (0 to L) where L represents the input paragraph length of input text, K_v and V_v are the input key and value, respectively. Finally, from Eq. (5) where $\hat{F}_{vt} = \{A_{vt}^0, A_{vt}^1, A_{vt}^2, \dots, A_{vt}^{L-1}\}$, we obtain the high level textual representation.

Aligning and merging the learned feature representations from the paragraph images and their ground truth is the final stage, accomplished through mutual self-attention. The Y_t transcription should align with the visual \hat{F}_{vt} output. The final prediction is achieved by first putting the visual representation \hat{F}_{vt} into a linear module and then into a softmax activation function in Eq. (6). It is anticipated that the output \hat{F}_{vt} will transcribe in according to the ground truth, Y_t . As a result, the final prediction is achieved by feeding the \hat{F}_{vt} to a softmax activation function in a linear module. Finally, the output prediction is generated using $softmax$ in Eq. (6).

$$O_t = softmax(\hat{F}_{vt}) \quad (6)$$

For instance, the anticipated transcription for the given input image is shown in Fig. 5. Precisely, to decode the letter "T" in the word "Template" from the first line of a given input paragraph image y_t , all characters of positions higher than the position of "T" are masked. This ensures that the decoding characters rely exclusively on predictions of the ordered sequence of "T" beforehand. When used in recurrent algorithms, the ability to analyze several time steps in parallel can significantly minimize training costs.

The proof text of bringing the Bible to the Temple is then quoted and the interpretation of the latter part of the verse by Rami ben Rabi is added, not because this is at all relevant to the discussion but because it was a familiar interpretation which had become so well known that it was invariably quoted whenever the verse itself was quoted, almost as if it were a part of the verse.



The proof text of bringing the Bible to the Temple is then quoted and the interpretation of the latter part of the verse by Rami ben Rabi is added, not because this is at all relevant to the discussion but because it was a familiar interpretation which had become so well known that it was invariably quoted whenever the verse itself was quoted, almost as if it were a part of the verse.



Fig. 3. Paragraph heat-map at decoder layer multi-head self-attention: randomly picked at 110 and 330 position character sequences.

4. Experiments

Experiments conducted on the paragraph level using the READ2016, RIMES, and IAM benchmark test datasets produced competitive results compared to recent models trained at the paragraph level, with less complexity. The ResNet-Transformer model's joint attention mechanism has effectively handled the challenges of recognizing offline handwritten paragraphs without explicit line segmentation.

4.1. Data sets

This research evaluated the proposed approaches on three well-known handwriting datasets: RIMES, IAM, and READ 2016. We used the paragraph levels for our purposes, with the commonly used split by researchers as detailed in Table 1.

4.1.1. IAM

The IAM [60] handwriting dataset is a collection of handwritten text documents widely used for research in handwriting recognition. It consists of more than 1500 pages of handwritten text, including various handwriting styles and languages. The documents in the dataset have been manually transcribed and annotated, making it a valuable resource for training and evaluating handwriting recognition algorithms. In our study, we utilized this dataset which consists of handwritten copies of text passages taken from the LOB corpus. The dataset includes grayscale images of English handwriting at a resolution of 300 dpi. It provides segmentation at the page, paragraph, line, and word levels and their corresponding transcriptions.

4.1.2. RIMES

The RIMES [61] handwriting dataset is a widely used collection of handwritten text documents for research in handwriting recognition. It consists of grayscale images of French handwritten text produced in the context of writing mail scenarios, with a resolution of 300 dpi. The official split has 1500 pages for training and 100 pages for evaluation. However, for comparison with other works, we utilized the last 100 training images for validation, as is commonly done. The dataset provides segmentation and transcription at the paragraph, line, and word levels, and we used paragraph segmentation levels in this study.

Table 1

IAM READ2016 and RIMES datasets associated with the standard three splits: train, valid, and test sets.

Dataset	Line-level			Paragraph-level		
	Train	Validation	Test	Train	Validation	Test
IAM	6482	972	2915	747	116	336
READ2016	8349	1040	1138	1584	179	197
RIMES	10,532	801	778	1400	100	100

4.1.3. READ2016

The READ2016 [62] handwriting dataset was proposed for the ICFHR 2016 competition on handwritten text recognition. It comprises a subset of the Ratsprotokolle collection used in the READ project and includes color images of Early Modern German handwriting. The dataset provides segmentation at the page, paragraph, and line levels. We follow the preprocessing steps of [34], so we eliminated the character 'r' from the ground truth as it is not genuine. The READ2016 dataset is frequently utilized with other datasets, such as the IAM and RIMES datasets, to enhance handwriting recognition systems' accuracy further.

4.2. Performance metrics

Character Error Rate (CER) is computed utilizing the Levenshtein distance [63]. This metric quantifies the number of character-level manipulations needed to transform the ground truth text into the Handwriting Text Recognition (HTR) output. The Levenshtein edit distance underpins the total count of insertions, replacements, and deletions required to transition from one sequence to another. Essentially, CER is delineated as expressed in Eq. (7).

$$CER = \frac{1}{|N|} \sum_{(p_i, l_i) \in N} LD(\hat{y}_i, y_i) \quad (7)$$

where $|N|$ is the number of ground truth characters at N partition, while the $LD(\hat{y}, y)$ is Levenshtein distance between prediction \hat{y} and target label y of i th character at each predicted character p_i with respect to the corresponding label l_i . Regarding WER in Eq. (8), it is defined similarly to CER. Whereas $LD(\hat{y}, y)$ is computed on word level, which requires transforming one string into another by dividing deletions D

words sum, insertions I , and substitutions S by a total number of the ground-truth N .

$$WER = \frac{S_{word} + I_{word} + D_{word}}{N_{words}} \quad (8)$$

4.3. Experimental setup

Fig. 1 describes the network architecture overview of the proposed solution for offline HTR on a paragraph level. The figure shows two network parts: the left part is the feature extraction module, and the right part is the encoder-decoder module. The left part of the network uses a convolutional neural network (CNN) with a split attention mechanism as the backbone for feature extraction. The split attention mechanism allows the network to focus on different parts of the input image simultaneously, improving the feature extraction process. The input image is a paragraph image, and the output of the CNN is a 2D feature representation. The right part of the network consists of an encoder and a decoder module, both using self-attention transformer layers. Before feeding the 2D feature representation to the encoder module, a 2D positional encoded technique is applied to the feature vector. The positional encoding helps the network to understand the spatial relationship between different parts of the input image. The decoder module also uses the same positional encoded technique for the embedded ground truth, which is the expected output of the network. The decoder module produces weighted masks that are utilized to calculate the probability distribution of each class with the help of a Softmax activation function. During training, the weights are updated using a Kullback-Leibler Divergence (KLD) loss function to reduce the loss on the subsequent epoch. The final feedforward layer generates predictions that form a probability distribution. The KLD metric compares this distribution with the sample distribution in the corresponding training dataset. The proposed solution uses a joint attention mechanism that combines vision and language models to achieve end-to-end recognition of handwritten paragraphs. The network architecture is straightforward to train and does not require line segmentation or a predefined lexicon model. The solution yields competitive results on benchmark datasets but requires much training data to be effective.

4.3.1. Training setup

Algorithm 1 Handwriting Paragraph Recognition (HTR) Model

Require: Input image I , target sequence T

- 1: $F \leftarrow \text{backbone}(I)$ {Extract features using the ResNeSt-50 backbone network}
- 2: $H \leftarrow \text{conv}(F)$ {Reduce feature dimensions}
- 3: Calculate positional encodings Pe using both sine (1) and cosine (2) frequency equations.
- 4: $H \leftarrow H + Pe$ {Add positional encodings to the input features}
- 5: Generate source mask M_S for H
- 6: $H \leftarrow \text{transformer_encoder}(H, M_S)$ {Pass features through the transformer encoder}
- 7: Generate target mask M_T and padding mask M_P for T
- 8: Calculate decoder embedding D and positional encoding Q for T
- 9: $O \leftarrow \text{transformer_decoder}(H, D + Q, M_T, M_P)$ {Pass features and target through the transformer decoder}
- 10: $Y \leftarrow \text{vocab}(O)$ {Predict output character probabilities}
- 11: **return** Y

Algorithm 1 shows the training procedures; we first extract the features from the input image: $F = \text{backbone}(I)$, followed by reducing the feature dimensions using the convolution layer: $H = \text{conv}(F)$, and calculating and adding positional encodings to the features: $H' = H + P$, then calculating decoder embedding and positional encoding for the target sequence: $T' = D(T) + Q(T)$. Finally, we pass the features and target sequences through the transformer decoder: $O =$

$\text{transformer_decoder}(H', T', M_T, M_P)$ to predict the output character probabilities: $Y = \text{vocab}(O)$. where: I represents the input image, F represents the feature map extracted from the input image using the backbone model, H represents the reduced feature dimensions after applying the convolution layer, P represents the positional encodings for the feature map, T represents the target sequence, $D(T)$ represents the decoder embedding of the target sequence, $Q(T)$ represents the positional encoding of the target sequence, M_T and M_P represent the target mask and padding mask, respectively, O represents the output of the transformer decoder and Y represents the output character probabilities.

The proposed method utilizes a ResNeSt50 convolutional neural network and a Transformer network to learn the underlying structure of input paragraph images and their corresponding transcriptions through supervised learning. During the training phase, the model is trained on a training dataset, while in the testing phase, the trained model weights are used to predict transcriptions for a testing dataset. The model is designed to take paragraph-level images as input and is trained end-to-end on a given dataset. The model weights are then saved based on the optimal training period and used by the paragraph recognition model to extract features. This method offers a powerful approach to handwriting paragraph recognition, leveraging convolutional and transformer-based architectures to capture complex features in the input images and corresponding transcriptions.

4.3.2. Cost function

During the optimization phase of our model training, we consistently evaluated errors by applying a corrective approach for mis-predictions. Selecting a suitable cost function for measuring a model performance is instrumental in adjusting weights to diminish loss in the subsequent training rounds. The problem dictates the type of loss function that should be used in neural network models. Our task uses the softmax activation function to establish a probability distribution for the multiclass classification problem.

To evaluate the performance of our handwriting paragraph recognition model, we employed the Kullback-Leibler Divergence (KLD) [64], a widely-used metric in information theory that measures the difference between two probability distributions. Specifically, the output of the final feedforward layer of our model was used to establish a probability distribution for each sample x . This distribution was then compared to the corresponding ground truth distribution for x from the training dataset. To quantify the discrepancy between the predicted distribution $P(i)$ and the ground truth distribution $G(i)$, we used KLD. The back-propagation operation was performed iteratively until the predicted distribution $P(i)$ yielded a textual transcription that closely matched or was identical to the ground truth distribution $G(i)$. We used the ADAM optimizer to modify the weights and biases of the model to achieve the most favorable distribution of prediction probabilities. The optimization process minimized the KLD between $P(i)$ and $G(i)$ as expressed in Eq. (9). Overall, the use of KLD allowed us to measure the accuracy of our handwriting paragraph recognition model by comparing the predicted distribution to the ground truth distribution for each sample in the training dataset. The iterative optimization process using KLD as a loss function enabled us to improve the model's accuracy and achieve better performance on the recognition task.

$$KLD(P||G) = - \sum_{i \in X} P(i) * \log\left(\frac{G(i)}{P(i)}\right) \quad (9)$$

In this context, i denotes a specific instance from the set of decoded ground truth elements of X and its corresponding encoded representation in the paragraph image features.

4.3.3. Regularization technique

Challenges such as overfitting and excessive confidence frequently arise when training deep learning models. Various regularization methods, including early stopping, weight decay, and dropout, have been developed to tackle the overfitting problem. However, the label smoothing technique [65] can effectively address both issues. Label smoothing, as depicted in Eqs. (10) and (11), assists in merging the uniform distribution with the updated one-hot-encoded label vector y , replacing the traditionally used label vector with this updated version.

$$\hat{y}_i = y_{hot}(1 - \alpha) + \frac{\alpha}{K} \quad (10)$$

In this equation, K is the total number of multi-class categories (97, 100, 89 for IAM, RIMES, and READ2016, respectively), and y_{hot} represents the embedded ground truth labels.

$$\hat{y}_i = \begin{cases} 1 - \alpha, & i = \text{target} \\ \alpha/K, & i \neq \text{target} \end{cases} \quad (11)$$

The label smoothing technique introduces a certain amount of noise to the distribution. This process effectively curtails the model tendency from extreme confidence in identifying the correct label. As a result, the disparity between the output values of predicted positive and negative samples is reduced, offering an effective strategy against overfitting and boosting the model capacity for generalization. We conducted a series of experiments and determined that an α value of 0.1 yielded the most optimal results.

4.3.4. Synthetic dataset and augmentation

We initially attempted to train our model using actual training data involving the best-selected hyperparameters. However, the model could not learn and converge due to data insufficiency. To overcome this problem, we decided to generate synthetic data since deep learning methods require a massive amount of labeled data to develop a generalized model. However, in the case of handwritten text recognition (HTR), there is a lack of adequate labeled data at the paragraph level. Therefore, we synthesized additional images by concatenating a random number of lines from the IAM, READ 2016, and RIMES datasets to expand these datasets. We generated 150,000 additional images to the initial 747, 1584, and 1400 paragraph forms on the IAM, READ 2016, and RIMES training sets, respectively. Fig. 4 shows a sample from the synthetic IAM data. This proves our model can transcribe handwritten text with strikethrough and circled words.

Unlike the original IAM dataset, the synthetic data are more challenging as they consist of random lines with variable lengths from the IAM dataset. To further enhance the training images, we utilized spontaneous augmentation approaches such as scaling, images, rotation, brightness, contrast, blurring, normal distribution, translation, and sharing.

5. Results and discussion

The methodology Section 3 is written rather clearly and provides many technical details. The experiment Section 4 adds further information about the system and training configuration and demonstrates the effectiveness of the proposed approach. In Section 4.2, we discussed an accurate assessment of the current study, using the evaluation metrics such as CER and WER. We directly compare our proposed model and the corresponding baselines (attention model without augmentation and non-attention model with augmentation) on the IAM, READ2016, and RIMES datasets. We use identical design decisions and training methodologies to ensure that our proposed joint attention-augmentation-based free performs comparably to the baselines. The following Sections 5.1, 5.2 briefly analyzed the reported results.

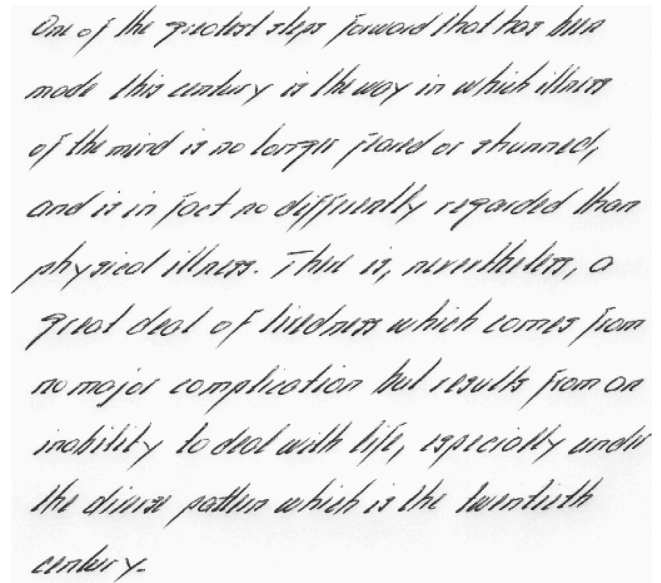


Fig. 4. A sample image of generated synthetic datasets from the IAM line dataset.

Table 2

The impact of the attention mechanism on IAM, READ 2016, and RIMES benchmark databases.

Model	CER (%)			WER (%)		
	IAM	Rimes	READ	IAM	Rimes	READ
No-attention	8.10	4.6	5.23	18.3	10.5	11.36
Attention	5.32	2.18	4.2	15.6	7.67	8.75

5.1. Quantitative analysis

The efficacy of an attention mechanism within the feature extraction framework is quantitatively demonstrated in Table 2. Initially, we utilized the pre-trained ResNet50 architecture for feature extraction, which lacked an attention mechanism. Conversely, ResNet50 incorporates an attention scheme, yielding more robust feature representations. Comparatively, the latter is dropping the CER and WER since the robust feature extractions are due to the split attention mechanism in its backbone block. Table 2 underscores the influence of the attention mechanism [66] across three benchmark datasets: IAM, READ 2016, and RIMES. The comparison includes two models: one without an attention mechanism and the other equipped with the attention mechanism. As mentioned, the standard key evaluation metrics used are CER and WER in our evaluation. The attention mechanism, a special layer of neural network technique, leads the model to pay more attention to specific aspects of input data resulting in enhanced recognition of offline handwritten paragraphs. The resnet-transformer model utilizes a joint attention mechanism, enabling implicit line segmentation and thus obviating the need for explicit line segmentation in text recognition. The table reveals the superior performance of the model incorporating the attention mechanism across all three benchmark datasets, with significantly lower CER and WER, which attests to the enhancement in model accuracy through the attention mechanism. The RIMES database shows the best performance, with the lowest CER and WER scores. In contrast, the IAM database presents the most significant challenge for the model due to its diversity of handwriting structures from more than 600 writers. In conclusion, the attention mechanism substantially elevates the accuracy of the model in recognizing offline handwritten paragraphs across all benchmark databases, with RIMES offering the best performance.

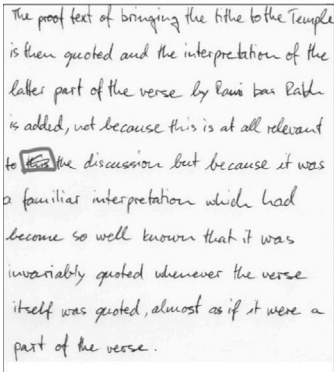
Input image	Ground Truth	Output Prediction
	The proof text of bringing the tithe to the Temple is then quoted and the interpretation of the latter part of the verse by Rami bar Rabbh is added, not because this is at all relevant to the discussion but because it was a familiar interpretation which had become so well known that it was invariably quoted the verse itself was quoted, almost as if it were a part of the verse.	The proof text of bringing the tithe to the Temple is then quoted and the interpretation of the latter part of the verse by Rami bar Rabbh is added, not because this is at all relevant to the discussion but because it was a familiar interpretation which had become so well known that it was invariably quoted the verge itself was quoted, almost as if it were a part of -he verge.

Fig. 5. Input image with the corresponding ground truth and its transcription from IAM dataset.

Table 3 provides a comprehensive performance comparison with the latest relevant research. Current models, as shown in the table, are divided into two approaches before the recognition phase: segmentation-free or segmentation-based paragraph recognition methods, as discussed in Section 2. We conducted three experiments. The first experiment involved a model without attention and with augmentation. The subsequent experiments using ResNeSt50 were performed either with augmentation (joint attention model with-aug) or without augmentation (joint attention model no-aug). The third experiment, devoid of split attention but with augmentation on ResNet50, is referred to as (our no-attention model with-aug). We also report the results of other methodologies in the literature that utilize both recurrent and non-recurrent models, with or without attention mechanisms.

Table 3 contrasts recent models that do not employ prior segmentation or language model correction for the IAM and RIMES datasets. This facilitates a more sound analysis of the effects of the split attention convolution network and transformer-based models relative to standard CNN and recurrent models. To the best of our knowledge, our proposed model secures the second-best performance on READ 2016, IAM, and RIMES benchmark datasets at paragraph levels with no line break pretraining as opposed to [34]. The recognition task was executed without needing prior segmentation or a lexicon model on both datasets. We achieved 4.2%, 5.62%, and 2.18% CER on READ2016, IAM, and RIMES, respectively, resulting in a slight 1% decrease in CER and WER compared to the most recent work on paragraphs by Denis et al. [34]. Although the results of [3] marginally surpass our CER by approximately 0.08% on the RIMES test set, their model was based on segmentation and used a lexicon model (LM) constraint to assist the BLSTM CTC decoder's predictions.

5.2. Qualitative analysis

We provide the complete ground truth and predictions for the image in Fig. 5. We present a selection of paragraph image predictions, illustrating the accuracy and reliability of our model, even when confronted with obstacles like overlapping characters and non-text noise. For instance, line 5 of the input image contains non-text noise, yet the predictions remain robust. Moreover, the model successfully handled an ambiguous non-textual element (highlighted with a manually drawn square) in the paragraph situated middle-left of Fig. 5. The model correctly predicted the text preceding and following this non-textual element, demonstrating its resilience to non-textual components in documents.

Fig. 3 showcases heatmaps of various positions in a paragraph at the decoder layer multi-head self-attention. We highlight character positions 110 and 330, showing that the model accurately locates these positions. The heatmaps support our hypothesis that the proposed model can competently predict full-page documents with varied context

layouts, encompassing text, non-text, diagrams, graphs, and other components. Despite the overall success, we acknowledge some challenges. Certain characters, such as “f,m,u, and b” were mispredicted due to significant overlap and intra-class variability amongst associated words. For example, the words “and” and “but” were erroneously predicted as “aud” and “lut”, respectively, as evidenced in the predictions under paragraph 5. However, these challenges can be addressed in future work by applying a lexicon and auto-correction models.

5.3. Comparison to the SOTA paragraph models

Table 3 compares the performance of the recent related models based on CER and WER across three benchmark datasets: RIMES, IAM, and READ 2016. The table presents a juxtaposition between the proposed joint attention model (with and without augmentation) and other contemporary models, including the vertical attention model, joint attention model with MDLSTM, a no-attention model with augmentation, HTR detection and transcription, scan-attend-read MDLSTM, joint attention MDLSTM, start-follow-read CNN-LSTM, and LexiconNet. CER and WER, critical metrics for evaluating handwriting recognition models, quantify the percentage of inaccurately recognized characters and words, respectively. Lower scores in both these metrics indicate superior model performance.

Table 3 also delineates two additional columns labeled ‘Lex’ and ‘Seg’, indicating the model is lexicon-based and segmentation-based, respectively. Our proposed joint attention model with augmentation surpasses most other models in CER and WER across all three datasets. The model does not necessitate explicit line segmentation, employing a split attention mechanism for implicit paragraph image segmentation instead. Our joint attention model with augmentation exhibits the lowest CER and WER scores for the IAM, Rimes, and READ2016 datasets compared to all other models in the table. The vertical attention model [34], while outperforming many other models, including our proposed joint attention-based approach, relies on a pre-trained line dataset and utilizes line break as a post-processing technique to strengthen model performance. LexiconNet [10], exhibiting competitive results, is reliant on [34] as a pre-trained model and employs lexicons, which may only sometimes be readily available or pragmatic for use.

6. Conclusion and future work

This study focuses on handwritten text recognition (HTR) at the paragraph level, introducing an end-to-end solution that leverages attention-based feature extraction and multi-head attention transformer-based encoder-decoder architecture. Our proposed architecture, which eliminates the need for line segmentation and a predefined lexicon model, streamlines the training process for the automatic

Table 3

Paragraph handwriting text recognition performance (CER, WER) using the ResNeSt-Transformer model compared to recent Handwriting Paragraph recognition models on the IAM, RIMES, and READ datasets.

Model	CER (%)			WER (%)			Lex	Seg
	IAM	Rimes	READ	IAM	Rimes	READ		
Vertical attention model (Line break) [34]	4.45	1.9	3.71	14.55	6.72	15.47	✗	✗
Joint attention model with-aug (Ours)	5.32	2.18	4.2	15.6	7.67	8.75	✗	✗
Joint attention model no-aug (Ours)	6.20	3.1	4.22	16.81	8.9	9.11	✗	✗
Joint attention-MDLSTM [25]	7.90	2.90	–	24.6	12.6	–	✗	✗
No-attention model with-aug (Ours)	8.10	4.6	5.23	18.3	10.5	11.36	✗	✗
HTR Detection and Transcription [44]	15.60	–	–	–	–	–	✗	✗
Scan-attend-read MDLSTM [26]	16.20	–	–	–	–	–	✗	✗
Scan-attend-read MDLSTM [26]	7.90	–	–	–	–	–	✗	✓
Scan-attend-read MDLSTM [26]	5.50	–	–	–	–	–	✓	✓
Joint attention MDLSTM [25]	6.50	–	–	–	–	–	✗	✓
Start-follow-read CNN-LSTM [3]	6.40	2.1	–	23.2	9.3	–	✓	✓
LexiconNet [10]	3.24	1.13	2.43	8.29	2.94	7.35	✓	✗

transcription of entire text paragraphs. Incorporating an attention mechanism within the feature extraction convolutional neural network (CNN) backbone (ResNeSt50) and the encoder–decoder Transformer-based, our model brings up joint attention across the backbone network and the encoder–decoder portion. Accordingly, the model performance is enhanced with the assistance of these attentive feature representations. Despite demonstrating competitive results on the IAM, READ2016, and RIMES datasets, our model requires considerable training data for optimal performance, which may constrain its practical application in specific contexts. However, our model can accurately transcribe image paragraphs of varying sizes, as substantiated by the qualitative result predictions.

Our future work will explore the potential of pre-processing techniques, such as recursive slanted text adjustment and illumination compensation, to enhance performance. Additionally, we plan to conduct experiments with and without label smoothing and various transformer variations to bolster model efficiency. Another promising avenue for further research that has emerged from our reviewers' feedback is the integration of a syntax checker within the output of our architecture. While the scope of the present work was primarily focused on the novel combination of ResNeSt50 with a transformer encoder–decoder model, adding a syntax checker could provide another layer of refinement specifically aimed at ensuring the grammatical correctness of the generated text. This enhancement would benefit commercial applications where high-quality, error-free output is crucial. We look forward to exploring this feature in our subsequent research. While not implemented in this study, this represents an exciting direction for future work, enabling us to continue improving our model's practical applicability and effectiveness. Future efforts will also extend our model capabilities to full-page recognition, accompanied by an in-depth analysis of the training process and optimal model selection. Moreover, we plan to broaden the generalizability of the model to accept input text in any script, facilitating efficient parallelization and reducing training time on large datasets.

CRedit authorship contribution statement

Mohammed Hamdan: Conceptualization, Methodology, Data analysis, Data processing, Literature review, Training analysis, Interpretation of results, Writing of the manuscript. **Mohamed Cheriet:** Overall critical revision of the manuscript and final approval of the manuscript.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request

Acknowledgments

The authors would like to thank NSERC, Canada for their financial support under grant # 2019-05230.

References

- [1] Graves A, Liwicki M, Fernández S, Bertolami R, Bunke H, Schmidhuber J. A novel connectionist system for unconstrained handwriting recognition. *IEEE Trans Pattern Anal Mach Intell* 2008;31(5):855–68.
- [2] Bianne-Bernard A-L, Menasri F, Mohamad RA-H, Mokbel C, Kermorvant C, Likforman-Sulem L. Dynamic and contextual information in HMM modeling for handwritten word recognition. *IEEE Trans Pattern Anal Mach Intell* 2011;33(10):2066–80.
- [3] Wigington C, Tensmeyer C, Davis B, Barrett W, Price B, Cohen S. Start, follow, read: End-to-end full-page handwriting recognition. In: *Proceedings of the European conference on computer vision. ECCV*, 2018, p. 367–83.
- [4] Inunganbi S. CP, K. M. Meitei Mayek handwritten dataset: compilation, segmentation, and character recognition. *Vis Comput* 2021;37:291–305.
- [5] Le AD. Automated transcription for pre-modern Japanese Kuzushiji documents by random lines erasure and curriculum learning. 2020, arXiv e-prints.
- [6] Yousef M, Hussain KF, Mohammed US. Accurate, data-efficient, unconstrained text recognition with convolutional neural networks. *Pattern Recognit* 2020;108:107482.
- [7] Dolfing HJGA. Whole page recognition of historical handwriting. 2020, arXiv e-prints.
- [8] Sharma A, Jayagopi DB. Towards efficient unconstrained handwriting recognition using dilated temporal convolution network. *Expert Syst Appl* 2021;164:114004.
- [9] Singh SS, Karayev S. Full page handwriting recognition via image to sequence extraction. 2021, p. 55–69.
- [10] Kumari L, Singh S, Rathore VVS, et al. LexiconNet: An end-to-end handwritten paragraph text recognition system. 2022, arXiv e-prints.
- [11] Coquenat D, Chatelain C, Paquet T. End-to-end handwritten paragraph text recognition using a vertical attention network. *IEEE Trans Pattern Anal Mach Intell* 2022;45:508–24.
- [12] Nag S, Ganguly PK, Roy S, et al. Offline extraction of indic regional language from natural scene image using text segmentation and deep convolutional sequence. 2018, p. 49–68.
- [13] Chowdhury A, Vig L. An efficient end-to-end neural model for handwritten text recognition. 2018, arXiv.
- [14] Carbonell M, Fornés A, Villegas M, et al. A neural model for text localization, transcription and named entity recognition in full pages. *Pattern Recognit Lett* 2020;136:219–27.
- [15] Bartz C, Seidel L, Nguyen D-H, et al. Synthetic data for the analysis of archival documents: Handwriting determination. 2020, p. 1–8.
- [16] Malik S, Sajid A, Ahmad A, et al. An efficient skewed line segmentation technique for cursive script OCR. *Sci Program* 2020;2020.
- [17] Kaur H, Kumar M. Offline handwritten Gurumukhi word recognition using extreme gradient boosting methodology. *Soft Comput* 2021;25:4451–64.
- [18] Peng D, Jin L, Ma W, et al. Recognition of handwritten Chinese text by segmentation: A segment-annotation-free approach. *IEEE Trans Multimedia* 2022;1.
- [19] Onim MdSH, Nyeem H, Roy K, Hasan M, Ishmam A, Akif MdAH, Ovi TB. BLNet: A new DNN model and Bengali OCR engine for automatic licence plate recognition. *Array* 2022;15:100244. <http://dx.doi.org/10.1016/j.array.2022.100244>.
- [20] Etaiwi W, Awajan A. SemanticGraph2Vec: Semantic graph embedding for text representation. *Array* 2023;17:100276. <http://dx.doi.org/10.1016/j.array.2023.100276>.

- [21] Minhuz Uddin Ahmed AJMd, Uddin MdA, Rahman MdA. Developing an algorithm for sequential sorting of discrete and connected characters using image processing of multi-line license plates. *Array* 2021;10:100063. <http://dx.doi.org/10.1016/j.array.2021.100063>.
- [22] Colter Z, Fayazi M, Youbi ZB-E, Kamp S, Yu S, Dreslinski R. Tabtext: A combined neural network and heuristic based table extractor. *Array* 2022;15:100220. <http://dx.doi.org/10.1016/j.array.2022.100220>.
- [23] Jayech K, Mahjoub MA, Amara NEB. Synchronous multi-stream hidden markov model for offline Arabic handwriting recognition without explicit segmentation. *Neurocomputing* 2016;214:958–71.
- [24] Roy PP, Bhunia AK, Pal U. Date-field retrieval in scene image and video frames using text enhancement and shape coding. *Neurocomputing* 2018;274:37–49.
- [25] Bluche T. Joint line segmentation and transcription for end-to-end handwritten paragraph recognition. *Adv Neural Inf Process Syst* 2016;29.
- [26] Bluche T, Louradour J, Messina R. Scan, attend and read: End-to-end handwritten paragraph recognition with mdlstm attention. In: 2017 14th IAPR international conference on document analysis and recognition, Vol. 1. ICDAR, IEEE; 2017, p. 1050–5.
- [27] Schall M, Schambach M-P, Franz MO. Multi-dimensional connectionist classification: Reading text in one step. In: 2018 13th IAPR international workshop on document analysis systems. DAS, IEEE; 2018, p. 405–10.
- [28] Yousef M, Bishop TE. OrigamiNet: weakly-supervised, segmentation-free, one-step, full page text recognition by learning to unfold. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020, p. 14710–9.
- [29] Puigcerver J. Are multidimensional recurrent layers really necessary for handwritten text recognition? In: 2017 14th IAPR international conference on document analysis and recognition (ICDAR), Vol. 1. IEEE; 2017, p. 67–72.
- [30] Hamdan M, Chaudhary H, Bali A, Cheriet M. Refocus attention span networks for handwriting line recognition. *Int J Docum Anal Recognit (IJDA)* 2022;1–17.
- [31] Naz S, Umar AI, Ahmad R, Ahmed SB, Shirazi SH, Siddiqi I, Razzak MI. Offline cursive Urdu–Nastaliq script recognition using multidimensional recurrent neural networks. *Neurocomputing* 2016;177:228–41.
- [32] Bluche T, Messina R. Gated convolutional recurrent neural networks for multilingual handwriting recognition. In: 2017 14th IAPR international conference on document analysis and recognition, Vol. 1. ICDAR, IEEE; 2017, p. 646–51.
- [33] Kaltenmeier A, Caesar T, Gloger JM, Mandler E. Sophisticated topology of hidden Markov models for cursive script recognition. In: Proceedings of 2nd international conference on document analysis and recognition (ICDAR'93). IEEE; 1993, p. 139–42.
- [34] Coquenat D, Chatelain C, Paquet T. End-to-end handwritten paragraph text recognition using a vertical attention network. *IEEE Trans Pattern Anal Mach Intell* 2022.
- [35] Sueiras J, Ruiz V, Sanchez A, Velez JF. Offline continuous handwriting recognition using sequence to sequence neural networks. *Neurocomputing* 2018;289:119–28.
- [36] Ma M, Wang Q-F, Huang S, Huang S, Goulermas Y, Huang K. Residual attention-based multi-scale script identification in scene text images. *Neurocomputing* 2021;421:222–33.
- [37] Graves A, Schmidhuber J. Offline handwriting recognition with multidimensional recurrent neural networks. *Adv Neural Inf Process Syst* 2008;21.
- [38] Papavassiliou V, Stafylakis T, Katsouros V, Carayannis G. Handwritten document image segmentation into text lines and words. *Pattern Recognit* 2010;43(1):369–77.
- [39] Graves A, Fernández S, Gomez F, Schmidhuber J. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: Proceedings of the 23rd international conference on machine learning. 2006, p. 369–76.
- [40] Mohamed M, Gader P. Handwritten word recognition using segmentation-free hidden Markov modeling and segmentation-based dynamic programming techniques. *IEEE Trans Pattern Anal Mach Intell* 1996;18(5):548–54.
- [41] Chen M-Y, Kundu A, Zhou J. Off-line handwritten word recognition using a hidden Markov model type stochastic network. *IEEE Trans Pattern Anal Mach Intell* 1994;16(5):481–96.
- [42] Tensmeyer C, Wigington C. Training full-page handwritten text recognition models without annotated line breaks. In: 2019 international conference on document analysis and recognition. ICDAR, IEEE; 2019, p. 1–8.
- [43] Chung J, Delteil T. A computationally efficient pipeline approach to full page offline handwritten text recognition. In: 2019 international conference on document analysis and recognition workshops, Vol. 5. ICDARW, IEEE; 2019, p. 35–40.
- [44] Carbonell M, Mas J, Villegas M, Fornés A, Lladós J. End-to-end handwritten text detection and transcription in full pages. In: 2019 international conference on document analysis and recognition workshops, Vol. 5. ICDARW, IEEE; 2019, p. 29–34.
- [45] Voigtlaender P, Doetsch P, Ney H. Handwriting recognition with large multi-dimensional long short-term memory recurrent neural networks. In: 2016 15th international conference on frontiers in handwriting recognition. ICFHR, IEEE; 2016, p. 228–33.
- [46] Wigington C, Stewart S, Davis B, Barrett B, Price B, Cohen S. Data augmentation for recognition of handwritten words and lines using a CNN-LSTM network. In: 2017 14th IAPR international conference on document analysis and recognition, Vol. 1. ICDAR, IEEE; 2017, p. 639–45.
- [47] Michael J, Labahn R, Grüning T, Zöllner J. Evaluating sequence-to-sequence models for handwritten text recognition. In: 2019 international conference on document analysis and recognition. ICDAR, IEEE; 2019, p. 1286–93.
- [48] Coquenat D, Chatelain C, Paquet T. Recurrence-free unconstrained handwritten text recognition using gated fully convolutional network. In: 2020 17th international conference on frontiers in handwriting recognition. ICFHR, IEEE; 2020, p. 19–24.
- [49] Moysset B, Kermorvant C, Wolf C. Full-page text recognition: Learning where to start and when to stop. In: 2017 14th IAPR international conference on document analysis and recognition, Vol. 1. ICDAR, IEEE; 2017, p. 871–6.
- [50] Khare V, Shivakumara P, Navya B, Swetha G, Guru D, Pal U, Lu T. Weighted-gradient features for handwritten line segmentation. In: 2018 24th international conference on pattern recognition. ICPR, IEEE; 2018, p. 3651–6.
- [51] Marti U-V, Bunke H. The IAM-database: an english sentence database for offline handwriting recognition. *Int J Docum Anal Recognit* 2002;5(1):39–46.
- [52] Grosicki E, Carre M, Brodin J-M, Geoffrois E. RIMES evaluation campaign for handwritten mail processing. 2008.
- [53] Zhang H, Wu C, Zhang Z, Zhu Y, Lin H, Zhang Z, Sun Y, He T, Mueller J, Manmatha R, et al. Resnest: Split-attention networks. 2020, arXiv preprint arXiv:2004.08955.
- [54] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. Attention is all you need. *Adv Neural Inf Process Syst* 2017;30.
- [55] Xie S, Girshick R, Dollár P, Tu Z, He K. Aggregated residual transformations for deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017, p. 1492–500.
- [56] Jeny AA, Sakib ANM, Junayed MS, Lima KA, Ahmed I, Islam MB. Sknet: A convolutional neural networks based classification approach for skin cancer classes. In: 2020 23rd international conference on computer and information technology. ICCIT, IEEE; 2020, p. 1–6.
- [57] Devlin J, Chang M-W, Lee K, Toutanova K. Bert: Pre-training of deep bidirectional transformers for language understanding. 2018, arXiv preprint arXiv:1810.04805.
- [58] Michael J, Labahn R, Grüning T, Zöllner J. Evaluating sequence-to-sequence models for handwritten text recognition. In: 2019 international conference on document analysis and recognition. ICDAR, IEEE; 2019, p. 1286–93.
- [59] Kang L, Toledo JI, Riba P, Villegas M, Fornés A, Rusinol M. Convoled, attend and spell: An attention-based sequence-to-sequence model for handwritten word recognition. In: German conference on pattern recognition. Springer; 2018, p. 459–72.
- [60] Marti U-V, Bunke H. The IAM-database: an english sentence database for offline handwriting recognition. *Int J Docum Anal Recognit* 2002;5:39–46.
- [61] Grosicki E, El-Abed H. Icdar 2011-french handwriting recognition competition. In: 2011 international conference on document analysis and recognition. IEEE; 2011, p. 1459–63.
- [62] Sanchez JA, Romero V, Toselli AH, Vidal E. ICFHR2016 competition on handwritten text recognition on the read dataset. In: 2016 15th international conference on frontiers in handwriting recognition. ICFHR, 2016, p. 630–5.
- [63] Konstantinidis S. Computing the levenshtein distance of a regular language. In: IEEE information theory workshop, Vol. 2005. IEEE; 2005, p. 4.
- [64] Moreno P, Ho P, Vasconcelos N. A Kullback-Leibler divergence based kernel for SVM classification in multimedia applications. *Adv Neural Inf Process Syst* 2003;16.
- [65] Müller R, Kornblith S, Hinton GE. When does label smoothing help? *Adv Neural Inf Process Syst* 2019;32.
- [66] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. 2014, arXiv preprint arXiv:1409.0473.