HAND: Hierarchical Attention Network for Multi-Scale Handwritten Document Recognition and Layout Analysis

Journal:	*Transactions on Pattern Analysis and Machine Intelligence*
Manuscript ID	TPAMI-2024-10-2922
Manuscript Type:	Regular (S1)
Keywords:	Handwritten document recognition, layout analysis, hierarchical attention, transformer decoder, postprocessing mT5, I.7 Document and Text Processing < I Computing Methodologies, Handwriting analysis ,Applications , Pattern Recognition

HAND: Hierarchical Attention Network for Multi-Scale Handwritten Document Recognition and Layout Analysis

Mohammed Hamdan, Abderrahmane Rahiche, *Member, IEEE,* Mohamed Cheriet, *Senior Member, IEEE,*

***Abstract*—Handwritten document recognition (HDR) is one of the most challenging tasks in the field of computer vision, due to the various writing styles and complex layouts inherent in handwritten texts. Traditionally, this problem has been approached as two separate tasks, handwritten text recognition and layout analysis, and struggled to integrate the two processes effectively. This paper introduces HAND (Hierarchical Attention Network for Multi-Scale Document), a novel end-to-end and segmentation-free architecture for simultaneous text recognition and layout analysis tasks. Our model's key components include an advanced convolutional encoder integrating Gated Depth-wise Separable and Octave Convolutions for robust feature extraction, a Multi-Scale Adaptive Processing (MSAP) framework that dynamically adjusts to document complexity and a hierarchical attention decoder with memory-augmented and sparse attention mechanisms. These components enable our model to scale effectively from single-line to triple-column pages while maintaining computational efficiency. Additionally, HAND adopts curriculum learning across five complexity levels. To improve the recognition accuracy of complex ancient manuscripts, we fine-tune and integrate a Domain-Adaptive Pre-trained mT5 model for post-processing refinement. Extensive evaluations on the READ 2016 dataset demonstrate the superior performance of HAND, achieving up to 59. 8% reduction in CER for line-level recognition and 31. 2% for page-level recognition compared to state-of-the-art methods. The model also maintains a compact size of 5.60M parameters while establishing new benchmarks in both text recognition and layout analysis. Source code and pre-trained models are available at https://github.com/MHHamdan/HAND.**

***Index Terms*—Handwritten document recognition, layout analysis, dual-path feature extraction, hierarchical attention, transformer decoder, post-processing mT5 model.**

I. Introduction

CONVERTING a digitized document image into a machine-readable format is a crucial process in the field of computer vision and natural language processing. The aim is to extract the content of document image scenes and its structure, enabling the identification and categorization of various elements within the document, such as text, images, and tables. This transformation not only facilitates efficient information retrieval and storage but also enhances the automation of document processing for subsequente applications, such as text analysis, summarization, translation, etc.

Handwritten document analysis presents unique challenges due to the variability in writing styles, complex layouts, and potential degradation of historical documents [1], [2]. Traditional approaches often separate the tasks of handwritten text recognition (HTR) and document layout analysis (DLA) [3], [4], leading to suboptimal results and increased computational complexity. This separation creates several challenges: first, errors in layout analysis can propagate to text recognition, affecting overall accuracy [5]; second, the sequential processing of these tasks increases computational overhead and processing time [6]; and third, the inability to leverage mutual information between layout and text features limits the model's ability to handle complex document structures [7]. Furthermore, these segregated approaches often struggle with historical documents where layout and text recognition are inherently interconnected due to varying writing styles, annotations, and structural degradation [8].

Recent advances in deep learning have paved the way for end-to-end approaches that can simultaneously handle both tasks, but significant challenges remain in processing multi-page documents, capturing long-range dependencies, and achieving high accuracy across diverse document structures. Recent attempts to address these limitations have shown promising but incomplete progress. While DAN [9] and Faster-DAN [10] introduced end-to-end approaches for document-level recognition, they remain limited to double-page columns and struggle with computational efficiency. DANCER [11] improved processing speed but still faces challenges with complex layouts and degraded historical texts. Current transformer-based models [12], [13] excel at character recognition but struggle with long-range dependencies in multi-page documents.

Furthermore, existing approaches often lack robust post-processing mechanisms for handling historical character variations and archaic writing styles [14], [15]. These limitations, combined with the computational challenges of processing large documents [16], [17], highlight the need for a more comprehensive and efficient solution that can handle the full spectrum of document complexity while maintaining high accuracy and reasonable computational

Authors are with Synchromedia laboratory, École de Technologie Supérieure (ÉTS), Montreal, Canada.
Manuscript received October XX, 2024; revised XX XX, 2025.

requirements.

To address these challenges, we propose the Hierarchical Attention Network for Multi-scale Document (HAND). As depicted in Fig. 1, our proposed end-to-end architecture for handwritten document recognition pushes the boundaries of current state-of-the-art models. HAND introduces several essential components that enable it to process complex, multi-page documents while maintaining high accuracy and efficiency.

The main contributions of our work are highlighted as follows:

- We introduce an end-to-end and segmentation-free architecture for handwritten document recognition for simultaneous text recognition and layout analysis tasks, scaling from single lines to triple-page columns.
- We propose a Multi-Scale Adaptive Processing (MSAP) framework that dynamically adjusts processing strategies based on document complexity, enabling efficient handling of diverse historical documents through memory-augmented attention and feature fusion.
- We design a dual-path encoder combining global and local features, enhanced with advanced convolutional layers, to effectively capture complex document layouts and handwriting styles in historical documents.
- We propose a hierarchical attention decoder with memory-augmented and sparse attention mechanisms, improving the model's ability to process large and complex documents efficiently.
- We implement adaptive feature fusion to balance fine-grained character details with broader structural patterns, enhancing overall document understanding.
- We incorporate a Domain-Adaptive Pre-trained mT5 model for post-processing in handwritten document recognition, which significantly improves accuracy for complex historical documents.
- We conduct extensive experiments on the READ 2016 dataset that demonstrate HAND's superior performance across various document scales, setting new state-of-the-art benchmarks in both text recognition and layout analysis while maintaining a compact model size.

The remaining parts of this paper are structured as follows: It starts with a review of existing work in Section II. The proposed HAND architecture is detailed in Section III, with its training strategy covered in Section IV. Post-processing techniques are discussed in Section V. Experimental results and discussions are provided in Section VI, and the paper concludes with future directions in Section VII.

II. Related Work

Handwritten Document Recognition (HDR) encompasses both text recognition and layout analysis for comprehensive manuscript processing. Fig. 2 illustrates the hierarchical complexity from line-level to triple-page. While it is not comprehensive, this section reviews relevant work in handwritten text recognition, document layout analysis, and recent advances in end-to-end HDR approaches.

A. Handwritten text recognition (HTR)

The text outlines the ubiquity of line-level text recognition in research, where each line is processed independently to identify characters and words. However, this approach may miss the contextual nuances present in multi-line or paragraph-level text. Current methods can be classified into segmentation-based and segmentation-free models.

1) Segmentation-based approaches: Where text recognition approaches follow a two-step process that first detects the text lines within a paragraph, and then recognizes the content of each line sequentially. Traditional approaches to HTR have focused on recognizing isolated text lines or words, requiring a prior line segmentation step. Various architectures have been proposed for this task, including Multi-Dimensional Long Short Term Memory (MD-LSTM) networks [18], combinations of Convolutional Neural Networks (CNN) and LSTM [20], and more recently, transformer-based models [12], [13], [23].

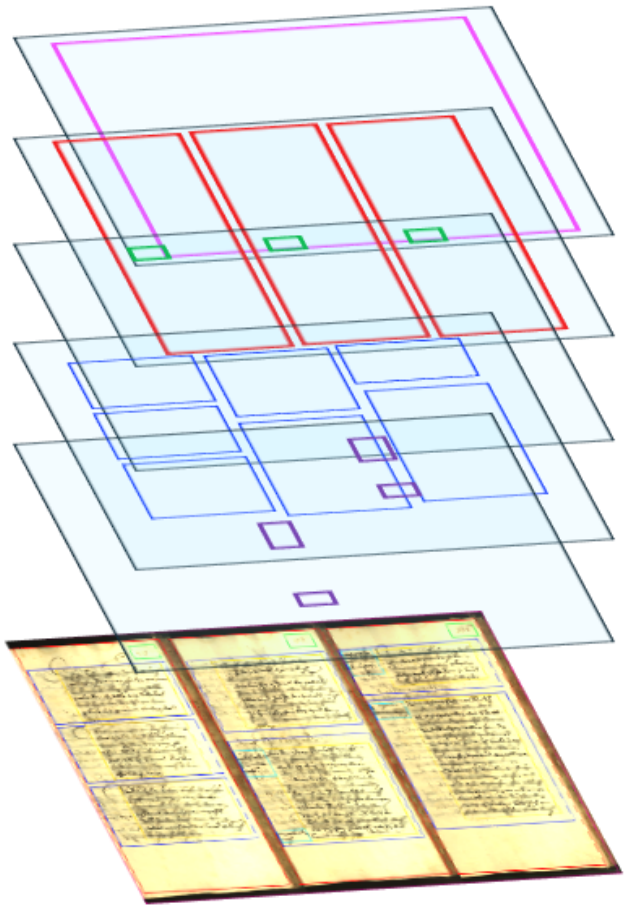
2) Segmentation-free approaches: To alleviate the need for line-level segmentation, some approaches have been developed to handle single-column pages or paragraphs. Yousef *et al.* [25] and Coquenet *et al.* [30] transformed the 2D image problem into a one-dimensional problem and used the Connectionist Temporal Classification (CTC) loss [31]. Bluche *et al.* [19] and Coquenet *et al.* [28] proposed attention-based models with a recurrent implicit line-segmentation process. More recent works have attempted to recognize handwritten content within paragraph images without explicit line segmentation [19], [21], [28]. Notably, Coquenet *et al.* [28] proposed the Vertical Attention Network (VAN), which uses an attention mechanism to select features representing the current text line being read, combined with an LSTM network.

B. Document layout analysis (DLA)

DLA aims to identify and categorize regions of interest in a document image. The field has evolved significantly, from traditional rule-based methods to modern deep learning approaches [32]. Current approaches can be broadly categorized into two main paradigms: pixel-based classification approaches and region-based methods [33].

1) Pixel-based approaches: Fully Convolutional Networks (FCN) are popular for pixel-level DLA [34]–[36]. These end-to-end models do not require rescaling input images and have been applied to various document types, including contemporary magazines, academic papers, and historical documents.

Recent advancements in pixel-based approaches have led to more sophisticated models. The study [37] proposed LayoutLM, a pre-training method that jointly models text and layout information in a unified framework, significantly improving performance on various document understanding tasks. Another notable work [38] introduced



(a) document hierarchical structures

```
<document>
  <page>
    <page_number>70</page_number>
    <section>
      <body>
        <segment>Text block ...</segment>
      </body>
    </section>
    <section>
      <body>
        <segment>Text block ...</segment>
        <annotation>
          <segment>Text block ...</segment>
        </annotation>
      </body>
    </section>
    <section>
      <body>
        <segment>Text block ...</segment>
      </body>
    </section>
  </page>
  <page>
    <page_number>93</page_number>
    <section>
      <body>
        <segment>Text block ...</segment>
      </body>
    </section>
    <section>
      <body>
        <segment>Text block ...</segment>
        <annotation>
          <segment>Text block ...</segment>
        </annotation>
        <annotation>
          <segment>Text block ...</segment>
        </annotation>
      </body>
    </section>
  </page>
  <page>
    <page_number>306</page_number>
    <section>
      <body>
        <segment>Text block ...</segment>
        <annotation>
          <segment>Text block ...</segment>
        </annotation>
      </body>
      <body>
        <segment>Text block ...</segment>
        <annotation>
          <segment>Text block ...</segment>
        </annotation>
      </body>
    </section>
  </page>
</document>
```

(b) XML representation

Fig. 1: Hierarchical recognition and organization of the content of a triple-page document image. The left side illustrates the hierarchical arrangement of layout elements. On the right, the corresponding XML structure.

a visual attention mechanism to improve the accuracy of layout analysis in historical documents.

2) Region-based approaches: Object detection approaches for word bounding box predictions have been studied by Carbonell et al. [39] and [40]. These methods follow the standard object-detection paradigm based on a region proposal network and non-maximum suppression algorithm.

Region-based approaches have seen significant improvements with the integration of deep learning techniques. For instance, [35] proposed dhSegment, a versatile deep-learning approach based on Fully Convolutional Networks (FCN) that can be applied to a variety of historical document processing tasks, including layout analysis. Authors [41] introduced a comprehensive framework for historical document layout analysis that combines region-based and pixel-based approaches to achieve robust performance across diverse document types.

C. Evolution and Current Challenges

Table 1 summarizes HDR development from line-level processing to complex document understanding. Early works [18], [20] established fundamental CNN-LSTM architectures, while later approaches [19], [25] introduced segmentation-free processing. Current state-of-the-art models, such as DAN [9], Faster-DAN [10], and DANCER [11], handle double-page documents in an end-to-end manner. However, they still suffer from several key limitations, as they generally struggle to scale beyond double-page documents and rely heavily on explicit segmentation annotations. Moreover, there is a clear separation of tasks between text recognition and layout analysis. Finally, these approaches often lack advanced post-processing techniques, particularly when dealing with historical texts. Thus, new research should focus on scaling to multi-page documents, integrating text and layout analysis, and incorporating advanced post-processing for historical manuscripts [42]–[44].

III. The proposed HAND

We describe the archictecture of the proposed HAND HAND, an end-to-end encoder-decoder architecture for joint text and layout recognition. As illustrated in Fig. 3, HAND's encoder comprises five advanced convolutional blocks that transform input document images into rich 2D feature maps $\boldsymbol{f}^{2D}$. These features, augmented with 2D

TABLE I: Comparison of Related Works in Terms of Task, Scalability, Context, and Segmentation-free Approach

Author	Task	Scalability	Context	Segmentation-free	Model Size (M)	Post-processing
Voigtlaender et al. (2016) [18]	HTR	Line	Global	No	-	No
Bluche et al. (2016) [19]	HTR	Paragraph	Global	Yes	-	No
Wigington et al. (2017) [20]	HTR	Line	Global	No	-	No
Bluche et al. (2017) [21]	HTR	Paragraph	Global	Yes (curriculum)	-	No
Coquenet et al. (2019) [22]	HTR	Line	Local	No	-	No
Michael et al. (2019) [23]	HTR	Line	Global	No	-	No
Kang et al. (2020) [12]	HTR	Line	Global	No	-	No
Coquenet et al. (2020) [24]	HTR	Line	Global	No	-	No
Yousef et al. (2020) [25]	HTR	Paragraph	Local	Yes	-	No
Wick et al. (2021) [13]	HTR	Line	Global	No	-	No
Li et al. (2021) [26]	HTR	Line	Global	No	-	No
Coquenet et al. (2021) [24]	HTR	Paragraph	Local	Yes	-	No
Singh et al. (2021) [27]	HTR + non-textual items	Page	Global	Yes (curriculum)	-	No
Rouhou et al. (2022) [14]	HTR + named entities	Paragraph	Global	Yes	-	No
Coquenet et al. (2022) [28]	HTR	Paragraph	Global	Yes	-	No
Hamdan et al. (2022) [29]	HTR	Line	Global	No	-	No
Hamdan et al. (2023) [15]	HTR	Paragraph	Global	Yes	-	No
Coquenet et al. (2022) [9]	HDR	Double-page	Global	Yes	7.03	No
Coquenet et al. (2023) [10]	HDR	Double-page	Global	Yes	7.03	No
Castro et al. (2024) [11]	HDR	Double-page	Global	Yes	6.93	No
This work (2024)	HDR	Triple-page	Global	Yes	5.60	Yes (mT5)

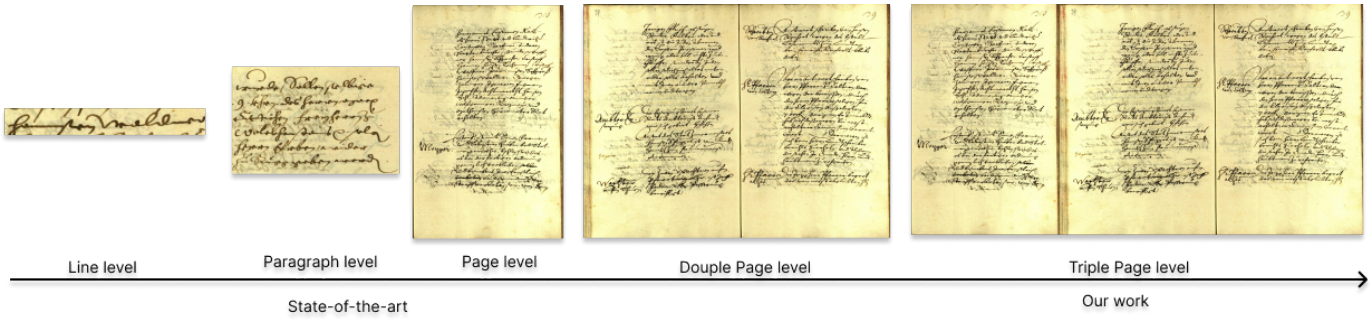


Fig. 2: Document recognition complexity across multiple scales: from line-level to triple-page documents. (a) Line level, (b) Paragraph level, (c) Single-page document, (d) Double-page document, (e) Triple-page document. Images are from the READ 2016 dataset.

positional encoding, are flattened into a 1D sequence $\boldsymbol{f}^{1D}$ for decoder processing.

A. Encoder

The HAND encoder is designed to efficiently extract multi-scale feature representations from input document images $\boldsymbol{X} \in \mathbb{R}^{H \times W \times 3}$, where H, W, and 3 represent the height, width, and number of channels (RGB), respectively. It generates feature maps $\boldsymbol{f}^{2D} \in \mathbb{R}^{H_f \times W_f \times C_f}$, where $H_f = \frac{H}{32}$. The width dimension W_f and number of channels C_f are defined as $\frac{W}{8}$, 64 for single-page inputs; $\frac{W}{16}$, 128 for double-page inputs; and $\frac{W}{32}$, 256 for triple-page inputs, respectively. Inspired by ResNet [45], the encoder starts with a large 7x7 kernel convolution to capture broad contextual information, gradually transitioning to 3x3 kernels in deeper layers for refined feature extraction. This approach balances global context with local details, crucial for understanding complex document layouts.

The encoder processes the input image with a Fully Convolutional Network (FCN) followed by a 2D Convolutional Layer (Conv2D), and generates a feature map $\boldsymbol{f}_1$. This can be formulated as follows:

$$\boldsymbol{f}_1 = \text{Conv2D}(\text{FCN}(\boldsymbol{X})). \quad (1)$$

This block aims to preserve the spatial information, that is crucial for document layout understanding, while allowing for local feature refinement, essential for character recognition [46].

To capture more information and intricacies from handwritten text, we apply a Gated Depth-wise Separable Convolution [47] to the obtained feature maps, allowing for deeper networks and more efficient representation. This process yields:

$$\boldsymbol{f}_2 = \sigma(\boldsymbol{W}_g * \text{DSConv}(\boldsymbol{f}_1)) \odot (\boldsymbol{W}_f * \text{DSConv}(\boldsymbol{f}_1)), \quad (2)$$

where σ is the sigmoid function, $\boldsymbol{W}_g$ and $\boldsymbol{W}_f$ are learnable weight matrices, $*$ denotes convolution, $\odot$ is element-wise multiplication, and DSConv is depth-wise separable convolution.

After that, we add an Octave Convolution [48] layer (OctaveConv) to decomposes the extarted features $\boldsymbol{f}_2$ into high and low frequency components, $\boldsymbol{f}_3^H$ and $\boldsymbol{f}_3^L$, respectively. This allows simultaneous capture of fine-grained character details and broader structural patterns within the document layout. The OctaveConv can be formulated as follows:

$$\boldsymbol{f}_3^H, \boldsymbol{f}_3^L = \text{OctaveConv}(\boldsymbol{f}_2). \quad (3)$$

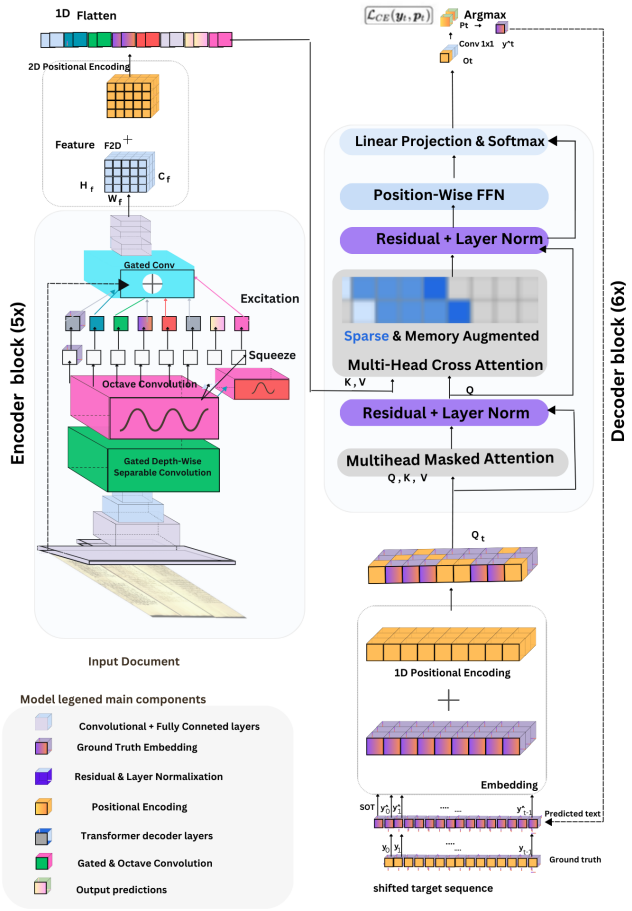


Fig. 3: Overview of the HAND Architecture: The HAND integrates convolutional layers as encoder for spatial feature extraction and a transformer decoder layers as a decoder for sequential prediction.

To enhance the network's ability to focus on informative features relevant to various handwriting styles and document layouts, we employ a Squeeze-and-Excitation (SE) [49] layer to adaptively recalibrate the channel-wise feature responses. This block fusions the two outputs of the previous layer into a calibrated feature $\boldsymbol{f}_4$.

$$\boldsymbol{f}_4 = \mathrm{SE}(\boldsymbol{f}_3^H, \boldsymbol{f}_3^L). \tag{4}$$

The next step involves Gated Convolution combined with an FCN:

$$\boldsymbol{f}_5 = \mathrm{FCN}(\mathrm{GatedConv}(\boldsymbol{f}_4)), \tag{5}$$

where $\boldsymbol{f}_5$ is the final output of this stage, GatedConv is the gated convolution operation. The gated convolution allows adaptive feature selection, crucial for handling the diverse characteristics of the input documents. The FCN maintains spatial coherence and enables dense predictions across the entire document [46], [50].

It is worth noting that after each convolutional operation, we apply Instance Normalization [51] and ReLU activation to introduce non-linearity. Instance Normalization was chosen over Batch Normalization to better handle the varying layouts and styles in document images. To enhance robustness, we implement a MixDropout strategy, combining standard dropout [52] and spatial dropout [53]. This technique improves the model's resilience against feature corruption, particularly beneficial for processing historical documents with varying degrees of degradation.

The final stage applies 2D positional encoding as defined in Eq. 6 before flattening the features into a 1D sequence. This encoding, adapted from the original transformer architecture [54], ensures that spatial information is preserved, which is essential for understanding complex document structures.

$$\boldsymbol{f}_j^{1D} = \mathrm{flatten}\left(\boldsymbol{f}_5^{2D} x, y + \mathrm{PE}^{2D}(x, y)\right), \tag{6}$$

where $j = y \cdot W_f + x$, mapping the 2D feature locations to 1D. The 2D positional encoding, $\mathrm{PE}^{2D}(x, y)$, is defined as:

$$\begin{aligned}
\mathrm{PE}^{2D}(x, y, 2i) &= \sin\left(\frac{x}{10000^{2i/d_\mathrm{m}}}\right) \\
\mathrm{PE}^{2D}(x, y, 2i+1) &= \cos\left(\frac{x}{10000^{2i/d_\mathrm{m}}}\right) \\
\mathrm{PE}^{2D}(x, y, 2i+d_\mathrm{m}/2) &= \sin\left(\frac{y}{10000^{2i/d_\mathrm{m}}}\right) \\
\mathrm{PE}^{2D}(x, y, 2i+1+d_\mathrm{m}/2) &= \cos\left(\frac{y}{10000^{2i/d_\mathrm{m}}}\right)
\end{aligned} \tag{7}$$

where, the sine and cosine functions are used to encode the positional information along both the x (horizontal) and y (vertical) axes of the document. The index i ranges over the feature dimensions, and d_model is the dimensionality of the model. This encoding ensures that each position in the 2D grid is uniquely represented, allowing the decoder to effectively attend to specific parts of the document based on their spatial location.

B. Decoder

The HAND decoder generates the output token sequence using a transformer-based approach, effectively capturing both local and global dependencies within the document through advanced attention mechanisms. It consists of 6 transformer decoder layers, inspired by the standard architecture [54], but with customization adapted for document recognition task.

The decoder begins by embedding previously generated output tokens into continuous vector representations, transforming each token y_{t-1} into a dense vector. Since transformers do not inherently capture sequence order, 1D Positional Encoding is added to the token embeddings, as defined in Equation 8, to inject positional information. This ensures the decoder retains awareness of token positions within the sequence, which is crucial for accurate sequence generation.

$$x_t = E(y_{t-1}) + \mathrm{PE}^{1D}(t) \tag{8}$$

where x_t is the input to the decoder at time step t, E is the embedding function, y_{t-1} is the previous output token, and PE^{1D} is the 1D positional encoding.

The decoder employs Masked Multi-Head Self-Attention to attend to past positions in the output sequence while

preventing access to future tokens, thereby enforcing causality. The mask matrix M, as defined in Equation 9, ensures that token predictions depend only on previously generated tokens, crucial for sequence generation tasks like text recognition.

$$\text{SelfAttn}(Q, K, V) = \sigma\left(\frac{QK^\top}{\sqrt{d_k}} + M\right)V \tag{9}$$

where Q, K, and V are the query, key, and value matrices respectively, d_k is the dimension of the keys, M is the mask matrix, and σ is the softmax function.

Following this, the Multi-Head Cross-Attention layer connects the decoder's current state with the encoded document features. This mechanism, defined in Equation 10, allows the decoder to selectively attend to the most relevant parts of the input document, enhancing its understanding of both local details and global context.

$$\text{CrossAttn}(Q, K, V) = \sigma\left(\frac{QK^\top}{\sqrt{d_k}}\right)V \tag{10}$$

where the variables are defined similarly to the self-attention equation.

To improve the model's ability to capture long-range dependencies, the decoder integrates Memory-Augmented Attention (Equation 11), inspired by memory networks [55]. This mechanism introduces a learnable memory matrix M, which stores global context information, enabling the decoder to maintain a broader understanding of document structure.

$$\mathcal{A}_M(Q, K, V, M) = \sigma\left(\frac{Q[K; M]^\top}{\sqrt{d_k}}\right)[V; M] \tag{11}$$

where M is the learnable memory matrix, and [;] denotes concatenation.

In parallel, Sparse Attention (Equation 12) is employed to enhance efficiency by focusing on critical local regions while reducing computational costs. Sparse Attention is particularly effective for processing long sequences, focusing on relevant tokens, making it beneficial for scaling documents to three columns.

$$\mathcal{A}_S(Q, K, V) = \sigma\left(\frac{QK^\top}{\sqrt{d_k}} \odot W\right)V \tag{12}$$

where W is a sparse attention mask.

The integration of memory-augmented and sparse attention mechanisms (Equation 13) ensures that the decoder can process relationships at different levels of granularity, effectively capturing both local details and broader dependencies within documents.

$$\text{Head}_i = \mathcal{A}_i(Q_i, K_i, V_i) \tag{13}$$

where $\mathcal{A}_i$ represents the combined attention mechanism for each head i.

To further enhance the decoder's adaptability, Adaptive Feature Fusion (Equation 14) selectively combines information across hierarchical levels, enabling the model to balance detailed and high-level representations. This mechanism captures both character-level and document-level features, crucial for comprehensive document understanding.

$$\mathbf{C} = \sum_l \lambda_l \cdot \text{MultiHeadAttn}_l(Q, K_l, V_l) \tag{14}$$

where $\mathbf{C}$ is the fused feature, λ_l are learnable weights, and l indexes over different levels of the hierarchy of $\boldsymbol{f}_j^{1D}$ feature map.

Residual connections and layer normalization, as defined in Equation 15, are used around each sub-layer to stabilize training and ensure efficient gradient flow, vital for training deep networks.

$$\boldsymbol{x}_{\text{out}} = \text{LayerNorm}(x_{\text{in}} + \text{Sublayer}(\boldsymbol{x}_{\text{in}})) \tag{15}$$

where x_{in} and x_{out} are the input and output of the sublayer, respectively.

After the attention layers, a Position-Wise Feed-Forward Network (PFFN) (Equation 16) introduces non-linearity and refines token-level representations, enhancing the model's ability to capture complex patterns.

$$\text{PFFN}(x) = \text{ReLU}(xW_1 + b_1)W_2 + b_2 \tag{16}$$

where W_1, W_2, b_1, and b_2 are learnable parameters.

The decoder concludes by generating output probabilities through a Linear Projection and Softmax operation (Equation 17), converting hidden states into a probability distribution over the target vocabulary.

$$p_t = \sigma(W_{\text{out}}h_t + b_{\text{out}}) \tag{17}$$

where p_t is the output probability distribution at time t, h_t is the hidden state, W_{out} and b_{out} are learnable parameters, and σ is the softmax function.

IV. Training Strategy

Training deep neural networks for document-level tasks presents unique challenges. While FasterDAN [10] introduced initial concepts for document-level recognition, we propose HAND with significant enhancements through adaptive processing and hierarchical learning. Our approach addresses three fundamental challenges: limited training data availability, absence of explicit segmentation labels, and efficient scaling across document complexities.

A. Pre-training and Enhancement Strategies

We establish robust feature extraction through pre-training with a CTC loss-based strategy:

$$\mathcal{L}_{\text{pre}} = -\log \sum_{t=1, \pi \in \mathcal{B}^{-1}(y)}^{2T} \prod_{t=1}^{T} p(\pi_t|x) \tag{18}$$

where $\mathcal{B}^{-1}(y)$ represents possible alignments between input x and target y. Pre-training uses 103 different fonts for data augmentation and employs curriculum dropout for robustness. We implement scheduled teacher forcing [56] with 20% random token errors and progressive dropout [57] (see Supplementary Material S.IV.).

B. Hierarchical Curriculum Learning

We implement a hierarchical curriculum learning strategy across five complexity levels: line, paragraph, page, double-page, and triple-page. The progressive learning schedule follows:

$$\mathcal{L}_{\text{curr}} = \sum_{l=1}^{L} \alpha_l \mathcal{L}_l(x_l, y_l) \tag{19}$$

where $L = 5$ represents curriculum levels, α_l is level-specific parameters, and $\mathcal{L}_l$ is the corresponding loss function.

Starting with line-level recognition on READ 2016 dataset, we progressively advance through multi-line blocks (spatial relationships), full pages (paragraphs, headers, annotations), double-page columns (cross-page relationships), and triple-page documents (long-range dependencies), as illustrated in Figure 5. Transfer learning initializes each level with previous weights:

$$W_l = W_{l-1} + \Delta W_l \tag{20}$$

To manage computational resources efficiently, we employ adaptive batch sizing [58]:

$$B_l = \max(\lfloor B_0 \cdot \gamma^l \rfloor, B_{min}) \tag{21}$$

where B_l is level-specific batch size, $\gamma < 1$ is decay factor, and B_{min} ensures minimum batch size.

We address data scarcity through synthetic data generation:

$$X_l^{syn} = G_l(z; \theta_l) \tag{22}$$

where G_l generates level-specific handwriting variations using 103 diverse historical fonts [59]. The generator reproduces authentic variations in slant, stroke width, and connectivity patterns, bridging the gap between synthetic and real manuscripts (see Supplementary Material S.IV. for detailed analysis).

C. Multi-Scale Adaptive Processing

HAND's effectiveness lies in its Multi-Scale Adaptive Processing (MSAP) framework, which enhances FasterDAN's [10] basic two-pass strategy through: (1) complexity-aware feature processing, (2) adaptive query generation, and (3) dynamic scale adaptation. Algorithm 1 orchestrates these components with encoder parameters θ_e, decoder parameters θ_d, and complexity network parameters ϕ.

The MSAP framework dynamically adjusts processing strategies based on document complexity through carefully designed adaptation mechanisms.

1) On-the-fly Complexity Assessment: Our dynamic complexity assessment mechanism operates for mapping document features to continuous scores:

$$C(x) = \phi(\text{Encoder}(x)) \in [0, 1] \tag{23}$$

The assessment employs a scale-aware architecture (see Supplementary Material S.III)

The complexity score guides feature selection:

$$F_s(x) = F(x) \odot \sigma(W_g[C(x); \text{Pool}(F(x))] + b_g) \tag{24}$$

Algorithm 1 HAND Training with MSAP

Input: $\mathcal{M}_{\text{line}}$, $L = 5$, E_l, α_0, ρ_0, G, B_0

$\theta_e \leftarrow \text{InitializeWeights}(\mathcal{M}_{\text{line}})$ {Initial encoder}
$\theta_d \leftarrow \text{RandomInitialize}()$ {Initial decoder}
$\phi \leftarrow \text{InitializeComplexityNetwork}()$ {Initial MSAP}
$W_0 \leftarrow \theta_e, \theta_d, \phi$ {Initial weights}
for $l \in L$ **do**
  $W_l \leftarrow W_{l-1} + \Delta W_l$ {Transfer learning (Eq. 20)}
  $\theta, \phi \leftarrow \text{AdaptParameters}(l, W_l)$ {Current level}
  $B_l \leftarrow \max(\lfloor B_0 \cdot \gamma^l \rfloor, B_{min})$ {Adapt batch(Eq. 21)}
  $X_l^{syn} \leftarrow G_l(z; \theta_l)$ {Synthetic data (Eq. 22)}
  $\mathcal{D}_l \leftarrow \text{PrepareDataset}(l, X_l^{syn})$ {Comb real synth}
  for $e = 1$ to E_l **do**
    $C_l \leftarrow \text{AssessComplexity}(\mathcal{D}_l, \phi)$
    $\alpha_l \leftarrow \text{ComputeCurriculumWeight}(l, e)$ {Eq. 19}
    for $b \in \mathcal{D}_l$ **do**
      $x_i' \leftarrow \text{ApplyAugmentation}(b)$
      $F_1 \leftarrow \text{FirstPassFeatures}(x_i', \theta, e)$ {Algo 2}
      $F_2 \leftarrow \text{SecondPassFeatures}(F_1, C_l)$ {Algo 3}
      $\mathcal{L}_l \leftarrow \text{ComputeLosses}(F_1, F_2, b)$
      $\mathcal{L}_{\text{total}} \leftarrow \alpha_l \mathcal{L}_l$ {Apply curriculum weight}
      $\theta, \phi \leftarrow \text{UpdateParameters}(\mathcal{L}_{\text{total}}, \theta, \phi)$
    end for
    $\alpha, \rho \leftarrow \text{AdjustHyperparameters}(e)$
  end for
  $W_l^* \leftarrow \{\theta, \phi\}$ {Save best model for current level}
end for
return W_{final} {Final HAND model parameters}

2) Adaptive Query Generation (AQG): AQG helps to effectively process documents of varying complexity through a two-pass approach. The first pass (Algorithm 2) focuses on extracting position-aware features while gradually incorporating spatial information during training. The second pass (Algorithm 3) generates complexity-aware queries that combine token-level, document-level, and positional information for effective attention computation.

In the first pass, position-aware feature extraction is achieved through:

$$f_1 = f_{\text{base}} + \alpha_e \text{PE}(f_{\text{base}}) \tag{25}$$

where f_{base} represents the initial encoder features and $\text{PE}(\cdot)$ denotes positional encoding. The modulation factor α_e ensures gradual incorporation of positional information through a warmup schedule:

$$\alpha_e = \alpha_0(1 + \gamma \min(1, e/E_{\text{warmup}})) \tag{26}$$

where $\alpha_0 = 0.1$ sets the initial positional influence, $\gamma = 0.5$ controls the integration rate, and $E_{\text{warmup}} = 150$ determines the warmup period. This creates three distinct phases: initial feature learning with minimal positional bias, gradual spatial information integration, and stable position-aware feature extraction.

In the second pass, adaptive query generation combines multiple information sources:

$$q_M^i = E(y_M^i) + \alpha_{C_l} P_{\text{doc}}^M + \beta_{C_l} R_M^i \tag{27}$$

Algorithm 2 First Pass Feature Extraction

Input: X, θ, e
Initialize: $E_{\text{warmup}} = 150$, $\alpha_0 = 0.1$, $\gamma = 0.5$
$\boldsymbol{f}_{\text{base}} \leftarrow \text{FCN}(\text{Conv2D}(X))$ {Initial features Eq.1}
$\boldsymbol{f}_{\text{gated}} \leftarrow \text{GatedDSConv}(f_{\text{base}})$ {Depth-wise Eq. 2}
$\boldsymbol{f}_{\text{oct}}^H, f_{\text{oct}}^L \leftarrow \text{OctaveConv}(f_{\text{gated}})$ {Freq./decom.Eq. 3}
$\boldsymbol{f}_{\text{se}} \leftarrow \text{SE}(f_{\text{oct}}^H, f_{\text{oct}}^L)$ {Channel recalib. Eq. 4}
$PE \leftarrow \text{PositionalEncoding2D}(f_{\text{se}})$ {Eq. 6}
$\alpha_e \leftarrow \alpha_0(1 + \gamma \min(1, e/E_{\text{warmup}}))$ {Eq. 26}
$\boldsymbol{f}_1 \leftarrow f_{\text{se}} + \alpha_e PE$ {Eq. 25}
return f_1

Algorithm 3 Second Pass Feature Processing

Input: $\boldsymbol{f}_1$, C_l, l, M
Initialize: $\lambda_{\text{mem}} = 0.5$, $\lambda_{\text{sparse}} = 0.5$
$\alpha_{C_l}, \beta_{C_l} \leftarrow \text{ComputeScalingFactors}(C_l)$ {Eq. 28}
$P_{\text{doc}}^M \leftarrow \text{DocumentContextEncoding}(f_1)$
$R_M^i \leftarrow \text{RelativePositionEncoding}(f_1)$
$q \leftarrow E(y_M^i) + \alpha_{C_l} P_{\text{doc}}^M + \beta_{C_l} R_M^i$ {Eq. 27}
$\omega_{C_l} \leftarrow \text{ComputeComplexityWeight}(C_l)$ { For Eq. 29}
$A_M \leftarrow \text{MemoryAugmentedAtten}(q, f_1, M)$ {Eq. 11}
$A_S \leftarrow \text{SparseAttention}(q, f_1)$ {Eq. 12}
$\boldsymbol{f}_2 \leftarrow \text{MultiHeadAtten}([A_M; A_S], f_1, \omega_{C_l})$ {Eq. 29}
$\boldsymbol{f}_2 \leftarrow f_2 \cdot \alpha_{C_l}$
return f_2

where $E(y_M^i)$ represents token embeddings for content understanding, P_{doc}^M captures document-level context, and R_M^i provides relative positional information. The complexity-dependent scaling factors α_{C_l} and β_{C_l} dynamically adjust the contribution of each component based on document complexity.

where, X represents the input document image, θ the encoder parameters, e the current epoch number, f_1 the first-pass features, C_l the document complexity score, l the curriculum level, and M the memory matrix for attention mechanisms. The initialization parameters E_{warmup}, α_0, and γ control the warmup schedule for positional encoding integration, while λ_{mem} and λ_{sparse} balance the contribution of memory-augmented and sparse attention mechanisms respectively.

3) Dynamic Scale Adaptation: Complexity-dependent scaling follows:

$$\begin{aligned} \alpha_{C_l} &= \alpha_0 \frac{1 + \gamma_{\alpha_{C_l}}}{1 + \exp(\delta_\alpha(C_l - \theta_\alpha))} \\ \beta_{C_l} &= \beta_0 \frac{1 + \gamma_{\beta_{C_l}}}{1 + \exp(\delta_\beta(C_l - \theta_\beta))} \\ \omega_{C_l} &= \omega_0 \frac{1 + \gamma_\omega C_l}{1 + \exp(\delta_\omega(C_l - \theta_\omega))} \end{aligned} \tag{28}$$

Our multi-head attention incorporates this complexity-aware weighting:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}} \cdot \omega(C_l)\right) V \tag{29}$$

The final attention output combines multiple heads:

$$\text{MultiHead}(Q, K, V, C_l) = \text{Concat}(\text{h}_1, ..., \text{h}_h) W^O \tag{30}$$

Where each head incorporates complexity-aware scaling:

$$\text{h}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V, C_l) \tag{31}$$

This integrated approach maintains essential attention capabilities while adaptively scaling according to document complexity (see Supplementary Material S.III. A).

D. Loss Function Design

Our training objective employs a novel adaptive loss weighting strategy that unifies layout understanding and text recognition. The total loss combines complexity-aware components:

$$\mathcal{L}_{\text{total}} = \lambda_{\text{layout}}(C_l)\mathcal{L}_{\text{layout}} + \lambda_{\text{text}}(C_l)\mathcal{L}_{\text{text}} + \lambda_c \mathcal{L}_c \tag{32}$$

For structured layout understanding and character recognition, we employ specialized loss functions:

$$\mathcal{L}_{\text{layout}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} y_{i,k}^{\text{layout}} \log(p_{i,k}^{\text{layout}}) \tag{33}$$

$$\mathcal{L}_{\text{text}} = -\frac{1}{N}\sum_{i=1}^{N}\frac{1}{T_i}\sum_{t=1}^{T_i}\sum_{v=1}^{V} y_{i,t,v}^{\text{text}} \log(p_{i,t,v}^{\text{text}}) \tag{34}$$

The complexity loss incorporates gradient regularization to ensure stable training:

$$\mathcal{L}_c = \underbrace{\|C(x) - C_{\text{target}}(x)\|_2^2}_{\text{MSE term}} + \underbrace{\lambda_{\text{reg}}\|\nabla_x C(x)\|_1}_{\text{gradient penalty}} \tag{35}$$

Dynamic weighting adapts to document complexity through sigmoid modulation:

$$\lambda_k(C_l) = \lambda_k^0 \cdot \frac{1 + \gamma_k C_l}{1 + \exp(\delta_k(C_l - \theta_k))} \tag{36}$$

This unified framework automatically balances layout and text recognition based on document complexity C_l, with enhanced focus on text recognition for simple documents and increased attention to layout understanding for complex documents. The complexity loss provides crucial supervision for adaptive processing strategies, enabling robust performance across varying document structures.

TABLE II: Layout recognition and additional metrics on the READ 2016 dataset without the PPER column.

Method	Single-Page		Double-Page		Triple-Page	
	mAP_{CER} ↑	LOER ↓	mAP_{CER} ↑	LOER ↓	mAP_{CER} ↑	LOER ↓
DAN [9]	93.32	5.17	93.09	4.98	–	–
Faster-DAN [10]	94.20	3.82	94.54	3.08	–	–
DANCER [11]	94.73	3.37	95.08	3.91	–	–
HAND	95.18	3.24	95.62	3.02	95.89	2.98
HAND+mT5	**95.76**	**3.11**	**96.14**	**2.89**	**96.35**	**2.85**

TABLE III: mT5 Post-Processing Impact

Scale	Error Rate		Improvement
	Base↓	+mT5↓	↑
Line	68.66%	5.31%	92.27%
Paragraph	89.21%	17.35%	80.55%
Single-Page	100%	42.29%	57.71%
Double-Page	100%	64.33%	35.67%
Triple-Page	100%	85.00%	15.00%

V. Post-Processing with mT5 for Error Correction

To enhance HAND's recognition accuracy for historical German manuscripts, we integrate a fine-tuned mT5-Small (300M parameters) model as a post-processing stage. Our adaptation process employs SentencePiece tokenization with custom layout tokens and Unicode NFKC normalization for historical character standardization [60]. For detailed implementation specifications, (see Supplementary Material S.V.). Through systematic analysis of HAND predictions after 1000 epochs on the READ 2016 dataset, we identified key recognition challenges, including similar-looking characters (ß→b), connected letters (ch→d), and archaic forms (s→f). These insights guided our data augmentation strategy, introducing variations in letter forms and spacing to reflect historical writing characteristics. For detailed performance analysis and optimization strategies, refer to section VI. We continuously evaluate performance using CER, WER, LOER, and mAP_{CER} metrics (Section VI-B), achieving improvements across all structural levels.

VI. Experimental Results

The evaluation of HAND's performance on the READ 2016 dataset addresses its capability to manage various document scales, from single lines to triple-page columns, to handle complex document structures and layouts.

A. The READ 2016 Dataset

We evaluate HAND on the READ 2016 dataset [61], which comprises historical documents from the *Ratsprotokolle* collection (1470-1805) of the Bozen state archive. These Early Modern German handwritten documents present significant challenges due to their complex layouts, diverse writing styles, and historical character variations. To evaluate our model's scalability, we utilize multiple variants of increasing complexity: from line-level to triple-page documents, as summarized in Table V. For comprehensive details about dataset characteristics, document scales, vocabulary distribution, and character set analysis, (see Supplementary Material S.V.I).

TABLE V: READ 2016 Dataset Statistics

Level	Train	Valid	Test	VocabSize
Line	8,367	1,043	1,140	89
Paragraph	1,602	182	199	89
Single-page	350	50	50	89
Double-page	169	24	24	89
Triple-page	112	15	15	89

B. Evaluation Metrics

We employ the following metrics to evaluate both text recognition and layout analysis capabilities. For text recognition, we use two unified formulations: one for character to paragraph level (Equation 37) and another for page-level documents (Equation 38).

$$\text{ER}_l = \frac{\sum_{i=1}^{k} d_{\text{lev}}(F_l(\hat{y}_i), F_l(y_i))}{\sum_{i=1}^{K} |F_l(y_i)|} \tag{37}$$

where ER_l is the Error Rate at level l, d_{lev} is the Levenshtein distance [62], and $F_l(\cdot)$ processes text according to level l. This formulation encompasses Character Error Rate (CER), Word Error Rate (WER), Sentence Error Rate (SER), and Paragraph Error Rate (PER).

$$\text{SPER}_n = \frac{\sum_{i=1}^{K} d_{\text{lev}}(G_n(\hat{y}_i), G_n(y_i))}{\sum_{i=1}^{K} |G_n(y_i)|} \tag{38}$$

where SPER_n measures error rates across n consecutive pages, used to compute Single-page (SPER), Double-page (DPER), and Triple-page (TPER) Error Rates.

For layout analysis, we adopt the Layout Ordering Error Rate (LOER, Equation 39) and Mean Average Precision (mAP_{CER}, Equation 40) metrics from [9]:

$$\text{LOER} = \frac{\sum_{i=1}^{K} \text{GED}(y_i^{\text{graph}}, \hat{y}_i^{\text{graph}})}{\sum_{i=1}^{K} (ne_i + nn_i)} \tag{39}$$

$$\text{mAP}_{\text{CER}} = \frac{\sum_{c \in S} \text{AP}_{\text{CER}}(c) \cdot \text{len}(c)}{\sum_{c \in S} \text{len}(c)} \tag{40}$$

C. Baseline Methods

We evaluate HAND against several state-of-the-art approaches across different document scales. For comprehensive evaluation, we group baselines into three categories based on their processing capabilities:

1) Line and Paragraph-level Baselines: At the line level, we compare against traditional CNN-based approaches like CNN+BLSTM [23] and CNN+RNN [61], which employ character-level attention mechanisms. We also include more recent transformer-based models such as ResneSt-Trans [15] that leverage paragraph-level attention. The FCN+LSTM approach [63] represents models with line-level attention mechanisms.

2) Page-level Baselines: For full-page processing, we primarily compare against three state-of-the-art end-to-end models: DAN [9], which pioneered end-to-end document-level recognition; Faster-DAN [10], which enhanced DAN's efficiency through optimized processing; and DANCER [11], which introduced improved processing speed for complex layouts. To the best of our knowledge, these are the only available models for such comparisons.

These models represent the current state-of-the-art in handling double-page documents but are limited in their ability to process triple-page columns. We include both the standard DANCER model and DANCER-Max, which employs additional optimization techniques.

TABLE IV: Computational efficiency comparison on the READ 2016 dataset

Model	#Params (M)	Single-Page		Double-Page		Triple-Page	
		Inf. Time (s)	Peak Mem (GB)	Inf. Time (s)	Peak Mem (GB)	Inf. Time (s)	Peak Mem (GB)
DAN [9]	7.03	3.55	8.9	8.50	9.0	-	-
FasterDAN [10]	7.03	0.66	9.1	1.90	9.3	-	-
DANCER [11]	6.93	0.51	3.5	0.87	3.3	-	-
HAND	**5.60**	**0.43**	**3.2**	**0.79**	**3.1**	**1.15**	**3.2**
HAND+mT5	5.60	63.15	11.95	140.25	12.03	202.35	15.02

3) Evaluation Results: Our experimental results demonstrate HAND's superior performance across all scales, as detailed in Tables VI and VII.

At the line level, HAND+mT5 achieves a CER of **1.65%** and WER of **4.95%**, representing **59.8%** and **71.9%** improvements over DAN's metrics (4.10% CER, 17.64% WER). For paragraph-level processing, HAND+mT5 attains a CER of **2.13%** and WER of **7.41%**, improving upon DAN by **33.8%** and **45.6%** respectively.

For page-level documents, HAND+mT5 demonstrates consistent improvements across all scales: For single-page documents, HAND+mT5 demonstrates a CER of **2.36%** and a WER of **9.73%**, showing reductions of 31.2%, 40.3%, and 29.8% compared to DAN, Faster-DAN, and DANCER respectively. For double-page documents, HAND+mT5 achieves a CER of **2.31%** and a WER of **8.22%**, achieving improvements of 37.6% over DAN (3.70%), 40.5% over Faster-DAN (3.88%), and 31.5% over DANCER-Max (3.37%). For triple-page documents, HAND+mT5 establishes a CER of **2.18%** and a WER of **6.03%**, establishing new benchmarks as the first model to effectively process triple-page columns.

TABLE VI: Line and paragraph recognition results on READ 2016

Method	Line		Paragraph	
	CER (%)	WER (%)	CER (%)	WER (%)
(CNN+BLSTM[a]) [23]	4.66	-	-	-
(CNN+RNN) [61]	5.1	21.1	-	-
(CNN+MDLSTM) [61]	4.8	20.9	-	-
(ResneSt-Trans[b]) [15]	-	-	4.20	8.75
(FCN+LSTM[c]) [63]	4.10	16.29	-	-
DAN [9]	4.10	17.64	3.22	13.63
HAND	2.71	10.98	3.18	12.55
HAND+mT5	**1.65**	**4.95**	**2.13**	**7.41**

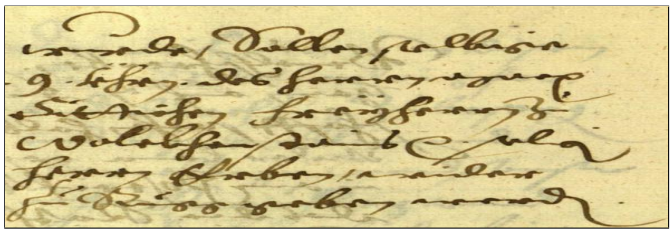
TABLE VII: Multi-scale recognition accuracy on READ

Method	Single-Page		Double-Page		Triple-Page	
	CER (%)	WER (%)	CER (%)	WER (%)	CER (%)	WER (%)
DAN [9]	3.43	13.05	3.70	14.15	-	-
Faster-DAN [10]	3.95	14.06	3.88	14.97	-	-
DANCER [11]	3.36	13.73	3.64	14.37	-	-
DANCER-Max [11]	3.37	13.00	3.37	13.34	-	-
HAND	3.41	13.79	3.46	13.58	3.52	13.82
HAND+mT5	**2.36**	**9.73**	**2.31**	**8.22**	**2.18**	**6.03**

D. Text recognition

To assess scalability, we conducted experiments at different levels, including line, paragraph, full page, and multi-page scales. Figure 4 illustrates a typical paragraph from the READ 2016 dataset that demonstrates the complexity of historical German handwriting.

HAND demonstrates consistent improvements over previous state-of-the-art models across all document scales and evaluation metrics. To further analyze the accuracy of HAND and HAND+mT5 predictions, we examine specific examples of error correction. The errors highlighted in red indicate misrecognized characters in HAND predictions, while improvements by HAND+mT5 are shown. For example, HAND incorrectly predicts "wurde" with a regular "u", while HAND+mT5 correctly recognizes it as "wūrde" with the proper diacritical mark. Additionally, HAND+mT5 improves the recognition of "widder" to "wider", demonstrating its capability to refine character-level predictions. These corrections showcase HAND+mT5's effectiveness in handling historical text characteristics, particularly diacritical marks and character variations. Tables VI and VII present a comprehensive comparison of HAND's performance against state-of-the-art models on the READ 2016 dataset across multiple document scales and evaluation metrics.



(a) Input image

(b) Ground truth (GT)	(c) HAND output	(d) HAND+mT5 output
wūrde, Sollen selbige .9. vhrn. des herrn Marx Sittichen Freyherrn Zū Wolckhenstains etc. selig herrn Erben, wider Zū Rūgg geben werd.	wurde, Sollen selbige .9. vhrn. des herrn Mavx Sittichen Freyhernn Zu Wolckhensteins etc. selig herrn Erbenn, widder Zu Rūgg geben word.	wūrde, Sollen selbige .9. vhrn. des herrn Marx Sittichen Freyharnn Zū Wolckhenstains etc. selig herrn Erbenn, wid-er Zū Rūgg geben werd.

Fig. 4: An exemple of handwritten text recognition using HAND, with (c) and without post-processing (d).

E. Layout analysis

We represent document structure through a hierarchical graph that captures both physical layout and logical organization (Fig. 5). The graph comprises nodes representing key document elements: Document (D) as the root node, Page (P) for individual pages, Section (S) for logical content segments, Page Number (N) for document ordering, Annotation (A) for supplementary information, and Body (B) for primary content across up to three columns. Directed edges encode flow and dependencies between elements. HAND demonstrates superior layout recognition capabilities across page scales. For single, double, and triple-page documents, HAND+mT5 achieves LOER scores of 3.11%, 2.89%, and 2.85% respectively,

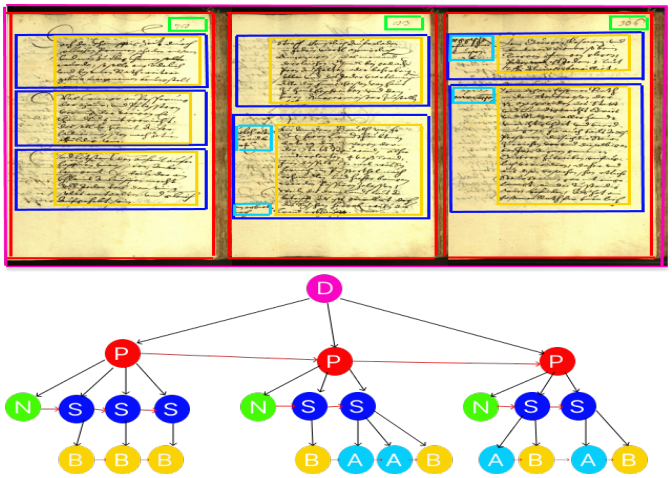


Fig. 5: Extended hierarchical structure of a triple-column document. Arrows indicate relationships among nodes: top-bottom indicates the hierarchical structure while left-right indicates the reading order.

significantly improving upon previous models such as DAN (5.17% single-page) and DANCER (3.37% single-page). The mean Average Precision (mAP_{CER}) shows consistent improvement, reaching 95.76%, 96.14%, and 96.35% for single, double, and triple-page documents respectively. Performance metrics improve with increasing document complexity. The CER progressively decreases from 2.36% (single-page) to 2.18% (triple-page), while mAP_{CER} increases from 95.76% to 96.35%, and LOER decreases from 3.11% to 2.85%. This scaling capability addresses a key limitation of previous models restricted to simpler document structures. HAND's combination of encoder-decoder architecture with pre-trained language model capabilities effectively leverages both visual and linguistic information, establishing new benchmarks in historical document processing.

F. Ablation Study

We conducted an extensive ablation study on the READ 2016 dataset to evaluate the contribution of each key component in HAND. Table VIII presents the results across different document scales after a 40-hour training period per configuration.

Each component proves critical for model performance. Removing data augmentation severely degrades accuracy across all scales (e.g., line-level CER increases from 1.65% to 25.89%). The absence of curriculum learning particularly impacts complex documents, with triple-page CER rising from 2.18% to 69.05%, aligning with DANCER's findings [11]. Synthetic data removal shows the most severe degradation (e.g., single-page CER rises to 79.39%), highlighting its importance for robust feature learning. The mT5 post-processing significantly enhances accuracy, as shown in Table III.

mT5's impact varies with document scale, showing strongest improvement at line level (92.27% reduction in error rate) and remaining significant but decreasing with increased document complexity (15.00% improvement for triple-page documents). These results demonstrate the synergistic effect of HAND's components in achieving state-of-the-art performance across scales.

G. Computational Efficiency

Comparing to FasterDAN two-pass strategy while integrating our MSAP framework (Section IV-C), HAND achieves significant computational efficiency improvements through complexity-aware processing. We evaluate HAND's computation performance against state-of-the-art models:

HAND achieves superior efficiency with 20% fewer parameters than previous models, demonstrating significant improvements in both inference time and memory usage. For single-page documents, HAND achieves **15.7%** faster inference with **8.6%** less memory than DANCER, while for double-page documents, it demonstrates **9.2%** faster inference with **6.1%** less memory. Notably, HAND+mT5 achieves highest accuracy but requires substantially more computational resources, presenting a clear trade-off between accuracy and efficiency. For detailed complexity analysis and computational considerations, (see Supplementary Material S.VII.).

VII. Conclusion

This paper introduced HAND, a novel architecture that advances HDR through three fundamental components. First, our advanced convolutional encoder provides robust feature extraction capabilities that effectively capture both fine-grained character details and broader structural patterns in historical documents. Second, the Multi-Scale Adaptive Processing (MSAP) framework revolutionizes document processing by dynamically adjusting to varying complexity levels, enabling efficient handling of documents from single lines to triple-column pages. Third, the hierarchical attention decoder with memory-augmented and sparse attention mechanisms has proven crucial in capturing both local character details and global document structures.

These three core components, working in concert with our curriculum learning strategy and mT5-based post-processing, have established new performance benchmarks across multiple scales. The model achieves significant improvements in recognition accuracy, with CER reductions of up to 59.8% for line-level and 31.2% for page-level recognition compared to previous state-of-the-art approaches. HAND maintains exceptional efficiency with just 5.60M parameters, demonstrating that sophisticated document understanding can be achieved with a relatively compact architecture.

Looking ahead, several promising research directions arise. First, the development of more efficient memory management techniques and model compression strategies could further optimize HAND's performance on very large

TABLE VIII: Ablation Study Results on READ 2016 Dataset

Configuration	Line		Paragraph		Single-Page		Double-Page		Triple-Page	
	CER	WER	CER	WER	CER	WER	CER	WER	CER	WER
Complete HAND	**1.65**	**4.95**	**2.13**	**7.41**	**2.36**	**9.73**	**2.31**	**8.22**	**2.18**	**6.03**
w/o Synthetic Data	29.92	48.45	46.18	63.83	79.39	92.77	75.01	91.11	75.12	91.23
w/o Curriculum	22.12	43.76	32.58	54.02	73.87	81.76	71.96	89.92	69.05	84.13
w/o Augmentation	25.89	46.74	37.37	56.28	75.65	83.21	72.72	90.38	89.81	14.62
w/o mT5	2.71	10.98	3.18	12.55	3.41	13.79	3.46	13.58	3.52	13.82

documents. Second, extending the model's multilingual capabilities through enhanced pre-training and adaptive tokenization would broaden its applicability across different historical manuscript and document collections. Finally, incorporating interpretability mechanisms could provide valuable insights into the model's decision-making process, particularly beneficial for historical document analysis where understanding recognition confidence is crucial.

HAND's success in combining state-of-the-art performance with architectural efficiency establishes a strong foundation for future advances in document processing, offering new possibilities for preserving and understanding our archival heritage.

Acknowledgments

The authors would like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC), Discovery Grant Program 05230-2019, for supporting this work.

References

[1] A. Fischer, V. Frinken, and H. Bunke, "Historical word-spotting in handwritten documents: The challenges," *International Conference on Frontiers in Handwriting Recognition*, pp. 106–111, 2012.

[2] T. Strauss, G. Leifert, T. Grüning, and R. Labahn, "A transformer-based neural network architecture for handwritten document analysis," *Pattern Recognition Letters*, vol. 136, pp. 187–195, 2020.

[3] S. S. Bukhari, T. M. Breuel, and F. Shafait, "Layout analysis for arabic historical document images using machine learning," *International Workshop on Historical Document Imaging and Processing*, pp. 130–137, 2012.

[4] S. Eskenazi, P. Gomez-Krämer, and J.-M. Ogier, "A comprehensive survey of mostly textual document segmentation algorithms since 2008," *Pattern Recognition*, vol. 64, pp. 1–14, 2017.

[5] S. Ahmed, M. I. Malik, M. Liwicki, and A. Dengel, "A survey on handwritten document understanding technique," *Pattern Recognition Letters*, vol. 94, pp. 39–57, 2016.

[6] C. Clausner, A. Hayes, and A. Antonacopoulos, "Efficient text line segmentation for historical documents," in *International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2019, pp. 723–728.

[7] C. Wei, E. Boudreau, and R. Singh, "End-to-end handwritten text recognition and word spotting with deep neural networks," *Pattern Recognition Letters*, vol. 129, pp. 158–165, 2020.

[8] F. Simistira, M. Seuret, N. Eichenberger, A. Garz, M. Liwicki, and R. Ingold, "Recognition of historical documents with few labeled samples," in *International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2017, pp. 251–255.

[9] M. Coquenet, C. Chatelain, and T. Paquet, "DAN: A segmentation-free document attention network for handwritten document recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 7, pp. 8229–8242, 2023.

[10] D. Coquenet, C. Chatelain, and T. Paquet, "Faster dan: Multi-target queries with document positional encoding for end-to-end handwritten document recognition," *arXiv preprint arXiv:2301.10593*, 2023.

[11] S. Castro, E. Vidal, and F. Casacuberta, "DANCER: A computationally efficient end-to-end model for handwritten document recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[12] L. Kang, P. Riba, M. Rusi nol, A. Forn'es, and M. Villegas, "Pay attention to what you read: Non-recurrent handwritten text-line recognition," *Pattern Recognition*, vol. 129, p. 108766, 2022.

[13] C. Wick, J. Z"ollner, and T. Gr"uning, "Transformer for handwritten text recognition using bidirectional post-decoding," in *International Conference on Document Analysis and Recognition*. Springer, 2021, pp. 112–126.

[14] A. C. Rouhou, M. Dhiaf, Y. Kessentini, and S. B. Salem, "Transformer-based approach for joint handwriting and named entity recognition in historical documents," *Pattern Recognition Letters*, vol. 155, pp. 128–134, 2022.

[15] M. Hamdan and M. Cheriet, "Resnest-transformer: Joint attention segmentation-free for end-to-end handwriting paragraph recognition model," *Array*, vol. 19, p. 100300, 2023.

[16] F. D. Keles, P. M. Wijewardena, and C. Hegde, "On the computational complexity of self-attention," in *International Conference on Algorithmic Learning Theory*. PMLR, 2023, pp. 597–619.

[17] Q. Fournier, G. M. Caron, and D. Aloise, "A practical survey on faster and lighter transformers," *ACM Computing Surveys*, vol. 55, no. 14s, pp. 1–40, 2023.

[18] P. Voigtlaender and H. Doetsch, Nay, "Handwriting recognition with large multidimensional long short-term memory recurrent neural networks," in *15th International Conference on Frontiers in Handwriting Recognition*. IEEE, 2016, pp. 228–233.

[19] T. Bluche, "Joint line segmentation and transcription for end-to-end handwritten paragraph recognition," in *Advances in neural information processing systems*, 2016, pp. 838–846.

[20] C. Wigington, S. Stewart, B. Davis, B. Barrett, B. Price, and S. Cohen, "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 (ICDAR)*, vol. 1. IEEE, 2017, pp. 639–645.

[21] T. Bluche, J. Louradour, and R. Messina, "Scan, attend and read: End-to-end handwritten paragraph recognition with mdlstm attention," *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1, pp. 1050–1055, 2017.

[22] D. Coquenet, Y. Soullard, C. Chatelain, and T. Paquet, "Have convolutions already made recurrence obsolete for unconstrained handwritten text recognition?" in *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, vol. 5. IEEE, 2019, pp. 65–70.

[23] J. Michael, R. Labahn, T. Gr"uning, and J. Z"ollner, "Evaluating sequence-to-sequence models for handwritten text recognition," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2019, pp. 1286–1293.

[24] D. Coquenet, C. Chatelain, and T. Paquet, "Recurrence-free unconstrained handwritten text recognition using gated fully convolutional network," in *2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 2020, pp. 19–24.

[25] M. Yousef and T. E. Bishop, "Origaminet: Weakly-supervised, segmentation-free, one-step, full page text recognition by learning to unfold," in *Proceedings of the conference on computer vision and pattern recognition*, 2020, pp. 14 710–14 719.

[26] M. Li, T. Lv, L. Cui, Y. Lu, D. Florencio, C. Zhang, Z. Li, and F. Wei, "Trocr: Transformer-based optical character recognition with pre-trained models," *arXiv arXiv:2109.10282*, 2021.

[27] S. Singh and S. Karayev, "Full page handwriting recognition via

image to sequence extraction," in *International Conference on Document Analysis and Recognition*. Springer, 2021, pp. 55–69.
[28] D. Coquenet, C. Chatelain, and T. Paquet, "End-to-end handwritten paragraph text recognition using a vertical attention network," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 1, pp. 508–524, 2023.
[29] M. Hamdan, H. Chaudhary, A. Bali, and M. Cheriet, "Refocus attention span networks for handwriting line recognition," *IJDAR*, vol. 26, no. 2, pp. 131–147, Jun. 2023.
[30] D. Coquenet, C. Chatelain, and T. Paquet, "Span: A simple predict and align network for handwritten paragraph recognition," in *International Conference on Document Analysis and Recognition*. Springer, 2021, pp. 70–84.
[31] A. Graves, S. Fern'andez, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd conference on Machine learning*, 2006, pp. 369–376.
[32] Y.-Y. Tang and C. Suen, "Recent progress in deep learning for historical document processing," *Pattern Recognition*, vol. 112, p. 107749, 2021.
[33] G. M. Binmakhashen and S. A. Mahmoud, "Document layout analysis: A comprehensive survey," *ACM Computing Surveys*, vol. 52, no. 6, pp. 1–36, 2019.
[34] X. Yang, E. Yumer, P. Asente, M. Kraley, D. Kifer, and C. L. Giles, "Learning to extract semantic structure from documents using multimodal fully convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4342–4351.
[35] S. A. Oliveira, B. Seguin, and F. Kaplan, "dhsegment: A generic deep-learning approach for document segmentation," in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 2018, pp. 7–12.
[36] Y. Soullard, P. Tranouez, C. Chatelain, S. Nicolas, and T. Paquet, "Multi-scale gated fully convolutional densenets for semantic labeling of historical newspaper images," *Pattern Recognition Letters*, vol. 131, pp. 435–441, 2020.
[37] Y. Xu, M. Li, L. Cui, S. Huang, F. Wei, and M. Zhou, "Layoutlm: Pre-training of text and layout for document image understanding," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 1192–1200.
[38] C. Soto and S. Yoo, "Visual attention for multi-task visual question answering," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1298–1307.
[39] M. Carbonell, A. Forn'es, M. Villegas, and J. Llad'os, "A neural model for text localization, transcription and named entity recognition in full pages," *Pattern Recognition Letters*, vol. 136, pp. 219–227, 2020.
[40] J. Chung and T. Delteil, "A computationally efficient pipeline approach to full page offline handwritten text recognition," in *2019 International Conference on Document Analysis and Recognition Workshops*, vol. 5. IEEE, 2019, pp. 35–40.
[41] L. Studer, M. Alberti, V. Pondenkandath, P. Goktepe, T. Kolonko, A. Fischer, and M. Liwicki, "A comprehensive study of document image layout analysis," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2019, pp. 1439–1446.
[42] R. R. Ingle, Y. Fujii, T. Deselaers, J. Baccash, and A. C. Popat, "A scalable handwritten text recognition system," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2019, pp. 17–24.
[43] J. C. A. Jaramillo, J. J. Murillo-Fuentes, and P. M. Olmos, "Boosting handwriting text recognition in small databases with transfer learning," in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 2018, pp. 429–434.
[44] R. Ptucha, F. P. Such, S. Pillai, F. Brockler, V. Singh, and P. Hutkowski, "Intelligent character recognition using fully convolutional neural networks," *Pattern recognition*, vol. 88, pp. 604–613, 2019.
[45] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.
[46] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2015, pp. 3431–3440.
[47] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1251–1258.
[48] Y. Chen, Y. Kalantidis, J. Li, S. Yan, and J. Feng, "Drop an octave: Reducing spatial redundancy in convolutional neural networks with octave convolution," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 3435–3444.
[49] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 7132–7141.
[50] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Free-form image inpainting with gated convolution," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 4471–4480.
[51] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," *arXiv preprint arXiv:1607.08022*, 2016.
[52] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.
[53] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 648–656.
[54] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.
[55] S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus, "End-to-end memory networks," in *Advances in neural information processing systems*, 2015, pp. 2440–2448.
[56] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," *Neural information processing systems*, vol. 28, 2015.
[57] P. Morerio, J. Cavazza, R. Volpi, R. Vidal, and V. Murino, "Curriculum dropout," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3544–3552.
[58] A. Devarakonda, M. Naumov, and M. Garland, "Adabatch: Adaptive batch sizes for training deep neural networks," *arXiv preprint arXiv:1712.02029*, 2017.
[59] "Browse Fonts - Google Fonts," May 2024, [Online; accessed 03. May. 2024]. [Online]. Available: https://fonts.google.com
[60] T. Kudo and J. Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2018, pp. 66–71.
[61] J. A. Sánchez, V. Romero, A. H. Toselli, and E. Vidal, "Icfhr2016 competition on handwritten text recognition on the read dataset," in *2016 15th International Conference on Frontiers in Handwriting Recognition*. IEEE, 2016, pp. 630–635.
[62] E. S. Ristad and P. N. Yianilos, "Learning string-edit distance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 522–532, 2002.
[63] D. Coquenet, C. Chatelain, and T. Paquet, "End-to-end handwritten paragraph text recognition using a vertical attention network," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 1, pp. 508–524, 2022.

Supplementary Material for "HAND: Hierarchical Attention Network for Multi-Scale Handwritten Document Recognition and Layout Analysis"

Mohammed Hamdan, Abderrahmane Rahiche, *Member, IEEE,* Mohamed Cheriet, *Senior Member, IEEE,*

S.I. Introduction

This document provides additional details and results that complement the main paper. Section S.III presents a detailed complexity analysis of multi-scale handwritten document recognition, examining spatial, sequential, and hierarchical complexities across different document scales. Section S.II introduces our Multi-Scale Adaptive Processing (MSAP) framework, detailing the complexity assessment mechanisms and adaptive query generation process. Section S.IV provides an in-depth analysis of our training framework, including hardware infrastructure, memory management, and context handling strategies. Section S.V offers comprehensive details about our mT5-based post-processing approach, including model adaptation, training procedures, and error correction mechanisms. Section S.VI presents a thorough analysis of the READ 2016 dataset characteristics and challenges. Finally, Section S.VII provides a detailed computational complexity study and resource allocation analysis.

S.II. Multi-Scale Adaptive Processing

The cornerstone of the proposed HAND effectiveness lies in its Multi-Scale Adaptive Processing (MSAP) framework. While FasterDAN [1] introduced a basic two-pass decoding strategy, our HAND architecture fundamentally enhances document recognition through three key innovations: (1) complexity-aware feature processing via the HAND encoder, (2) adaptive query generation, and (3) dynamic scale adaptation through MSAP. As shown in Algorithm hand training framework, these components work in concert to handle documents ranging from simple line-level texts to complex multi-page layouts, with θ_e and θ_d representing the HAND encoder and decoder parameters respectively. The MSAP framework is integrated through the complexity network parameters ϕ, enabling dynamic processing adjustments based on document structure complexity.

Unlike fixed processing pipeline in FasterDAN model, our MSAP framework dynamically adjusts its processing strategy based on document structure complexity. Algorithm HAND training presents our unified training framework, which orchestrates these components through carefully designed adaptation mechanisms.

Central to our adaptive processing is a dynamic complexity assessment mechanism that maps document features to continuous complexity scores during training:

$$C(x) = \phi(\text{Encoder}(x)) \in [0, 1] \tag{S.1}$$

where ϕ assessment defined using (Eq.S.2) and applied at the beginning of each training batch and implements a scale-aware architecture designed to capture both local and global document characteristics.

$$\begin{aligned} \boldsymbol{f}_{\text{pool}} &= \text{AdaptivePool}(f, [H_l, W_l]) \\ h_1 &= \text{LayerNorm}(\text{ReLU}(W_1 f_{\text{pool}} + b_1)) \\ h_2 &= \text{Dropout}(p_d) \cdot \text{ReLU}(W_2 h_1 + b_2) \\ C(x) &= \sigma(W_3 h_2 + b_3) \end{aligned} \tag{S.2}$$

where H_l and W_l are adaptively determined based on the input level as $[4, 16]$, $[8, 16]$, and $[16, 16r]$ for line, paragraph, and page levels respectively. Here, r represents the aspect ratio and scales with the page width, adopting values of 1, 2, or 3 based on the scale level.

The architecture (Eq. S.2 employs several key components working in concert. AdaptivePool dynamically adjusts pooling dimensions based on input scale, enabling consistent processing from single lines to triple-page spreads. Feature projection, defined as $W_1 \in \mathbb{R}^{d_h \times (H_l W_l d_f)}$, projects pooled features into a high-dimensional hidden space. Normalization is achieved through LayerNorm, which stabilizes feature distributions across different document types. Regularization is enforced by using Dropout with $p_d = 0.2$ to prevent overfitting to specific layout patterns. Non-linear transformations are made possible through matrices W_2 and W_3, enabling crucial transformations for capturing complex document structures.

The resulting complexity score guides subsequent processing through our novel dynamic feature selection mechanism:

$$F_s(x) = F(x) \odot \sigma(W_g[C(x); \text{Pool}(F(x))] + b_g) \tag{S.3}$$

where the concatenation $[C(x); \text{Pool}(F(x))]$ combines global complexity assessment with local feature characteristics. By automatically detecting the current processing level— at the line, paragraph, or page scale— ϕ guides resource allocation, computational strategy selection, and balances attention between local character details and global layout structure, while also facilitating smooth

transitions between curriculum learning levels, thereby enabling our model to efficiently handle documents of varying complexity and maintain optimal resource utilization throughout training.

A. Adaptive Query Generation

In contrast to the sequential processing in FasterDAN, our pipeline operates through two coordinated passes integrated within the MSAP framework (Algorithm hand training), lines 16-17). Each pass is optimized for different aspects of document understanding:

The first pass, detailed in Algorithm first pass, implements position-aware feature extraction with adaptive feature enhancement:

$$\boldsymbol{f}_1 = \boldsymbol{f}_{\text{base}} + \alpha(e)\text{PE}(\boldsymbol{f}_{\text{base}}) \tag{S.4}$$

where $\boldsymbol{f}_{\text{base}}$ represents the initial encoder features and $\text{PE}(\cdot)$ denotes positional encoding. The modulation factor $\alpha(e)$ ensures gradual incorporation of positional information through a warmup schedule:

$$\alpha(e) = \alpha_0(1 + \gamma \min(1, e/E_{\text{warmup}})) \tag{S.5}$$

where $\alpha_0 = 0.1$ sets the initial positional influence, $\gamma = 0.5$ controls the rate of positional encoding integration, and $E_{\text{warmup}} = 150$ determines the warmup period. This configuration creates three distinct training phases: (1) initial feature learning with minimal positional bias ($\alpha \approx 0.1$) during the first few epochs, allowing the model to focus on basic feature extraction; (2) gradual integration of spatial information as α increases to 0.15 over the 150-epoch warmup period, helping the model learn position-aware features without destabilizing training; and (3) stable position-aware feature extraction post-warmup. This adaptive scheme proves particularly effective for our hierarchical document understanding tasks, where spatial relationships become increasingly important as we progress from line-level to multi-page document processing.

The second pass, detailed in Algorithm (alg second pass), introduces our novel Multi-Scale Adaptive Query (MSAQ) mechanism where the key contribution is the adaptive query generation:

$$q_M^i = E(y_M^i) + \alpha(C_l)P_{\text{doc}}^M + \beta(C_l)R_M^i \tag{S.6}$$

Equation S.14 represents a query mechanism that integrates three main components. Token embeddings $E(y_M^i)$ for content understanding, document-level context P_{doc}^M for structural awareness, and relative positional information R_M^i for spatial relationships.

The integration of the first pass (Algorithm first pass) and the second pass (Algorithm second pass) into the hierarchical curriculum learning framework is structured through the main training loop (Algorithm hand training). The complexity score C_l, calculated in the main algorithm at line 12, serves as a pivotal factor influencing both the feature enhancement in the first pass and the adaptive query processing in the second pass. This integration is designed to ensure that feature extraction is adaptable to each curriculum level, query processing is capable of scaling with document complexity, and attention mechanisms are optimized to align with the current learning stage.

B. Dynamic Scale Adaptation

The complexity-dependent scaling employs sophisticated adaptation:

$$\begin{aligned}\alpha(C_l) &= \alpha_0 \frac{1 + \gamma_\alpha C_l}{1 + \exp(\delta_\alpha(C_l - \theta_\alpha))} \\ \beta(C_l) &= \beta_0 \frac{1 + \gamma_\beta C_l}{1 + \exp(\delta_\beta(C_l - \theta_\beta))}\end{aligned} \tag{S.7}$$

The parameters γ_α, γ_β, θ_α, θ_β, δ_α, and δ_β control complexity sensitivity, are learned thresholds, and determine the transition sharpness, respectively.

Further extending beyond FasterDAN, our multi-head attention mechanism incorporates complexity-aware weighting that adapts the cross-attention mechanism from our decoder architecture:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}} \cdot \omega(C_l)\right)V \tag{S.8}$$

where $\omega(C_l)$ is a learned complexity-dependent attention scaling function that modulates the base attention mechanism Equation cross attention according to document complexity. This scaling ensures that the attention weights adapt to both the local token-level features and global document structure based on the complexity assessment determined by (Equation S.1).

The interaction between the decoder's standard attention mechanism and this complexity-aware scaling produces the final attention output:

$$\text{MultiHeadAttn}(Q, K, V, C_l) = \text{Concat}(\text{h}_1, ..., \text{h}_h)W^O \tag{S.9}$$

where each attention head h_i incorporates complexity-aware scaling:

$$\text{h}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V, C_l) \tag{S.10}$$

The integrated approach (Equation S.18) enables our model to not only maintain the HAND decoder's essential attention capabilities for accurate sequence modeling but also to adaptively scale attention according to the complexity of documents subsubsec complexity assessment as discussed in the paper. It ensures a balance between local character recognition and a comprehensive understanding of the global layout, allowing for a smooth transition between different processing document levels. For a detailed analysis of HAND's context utilization compared to existing approaches, see Supplementary Material A.

S.III. Complexity Analysis of Multi-Scale HDR

The cornerstone of the proposed HAND effectiveness lies in its Multi-Scale Adaptive Processing (MSAP) framework. While FasterDAN [1] introduced a basic two-pass decoding strategy, our HAND architecture fundamentally enhances document recognition through three key innovations: (1) complexity-aware feature processing via the

HAND encoder, (2) adaptive query generation, and (3) dynamic scale adaptation through MSAP. As shown in Algorithm hand training framework, these components work in concert to handle documents ranging from simple line-level texts to complex multi-page layouts, with θ_e and θ_d representing the HAND encoder and decoder parameters respectively. The MSAP framework is integrated through the complexity network parameters ϕ, enabling dynamic processing adjustments based on document structure complexity.

Unlike fixed processing pipeline in FasterDAN model, our MSAP framework dynamically adjusts its processing strategy based on document structure complexity. Algorithm HAND training presents our unified training framework, which orchestrates these components through carefully designed adaptation mechanisms.

The architecture (Equation S.2 employs several key components working in concert. AdaptivePool dynamically adjusts pooling dimensions based on input scale, enabling consistent processing from single lines to triple-page spreads. Feature projection, defined as $W_1 \in \mathbb{R}^{d_h \times (H_l W_l d_f)}$, projects pooled features into a high-dimensional hidden space. Normalization is achieved through LayerNorm, which stabilizes feature distributions across different document types. Regularization is enforced by using Dropout with $p_d = 0.2$ to prevent overfitting to specific layout patterns. Non-linear transformations are made possible through matrices W_2 and W_3, enabling crucial transformations for capturing complex document structures.

The resulting complexity score guides subsequent processing through our novel dynamic feature selection mechanism:

$$F_s(x) = F(x) \odot \sigma(W_g[C(x); \text{Pool}(F(x))] + b_g) \quad \text{(S.11)}$$

where the concatenation $[C(x); \text{Pool}(F(x))]$ combines global complexity assessment with local feature characteristics. By automatically detecting the current processing level— at the line, paragraph, or page scale— ϕ guides resource allocation, computational strategy selection, and balances attention between local character details and global layout structure, while also facilitating smooth transitions between curriculum learning levels, thereby enabling our model to efficiently handle documents of varying complexity and maintain optimal resource utilization throughout training.

A. Adaptive Query Generation

In contrast to the sequential processing in FasterDAN, our pipeline operates through two coordinated passes integrated within the MSAP framework (Algorithm hand training), lines 16-17). Each pass is optimized for different aspects of document understanding:

The first pass, detailed in Algorithm first pass, implements position-aware feature extraction with adaptive feature enhancement:

$$f_1 = f_{\text{base}} + \alpha(e)\text{PE}(f_{\text{base}}) \quad \text{(S.12)}$$

where f_{base} represents the initial encoder features and $\text{PE}(\cdot)$ denotes positional encoding. The modulation factor $\alpha(e)$ ensures gradual incorporation of positional information through a warmup schedule:

$$\alpha(e) = \alpha_0(1 + \gamma \min(1, e/E_{\text{warmup}})) \quad \text{(S.13)}$$

where $\alpha_0 = 0.1$ sets the initial positional influence, $\gamma = 0.5$ controls the rate of positional encoding integration, and $E_{\text{warmup}} = 150$ determines the warmup period. This configuration creates three distinct training phases: (1) initial feature learning with minimal positional bias ($\alpha \approx 0.1$) during the first few epochs, allowing the model to focus on basic feature extraction; (2) gradual integration of spatial information as α increases to 0.15 over the 150-epoch warmup period, helping the model learn position-aware features without destabilizing training; and (3) stable position-aware feature extraction post-warmup. This adaptive scheme proves particularly effective for our hierarchical document understanding tasks, where spatial relationships become increasingly important as we progress from line-level to multi-page document processing.

The second pass, detailed in Algorithm (alg second pass), introduces our novel Multi-Scale Adaptive Query (MSAQ) mechanism where the key contribution is the adaptive query generation:

$$q_M^i = E(y_M^i) + \alpha(C_l)P_{\text{doc}}^M + \beta(C_l)R_M^i \quad \text{(S.14)}$$

Equation S.14 represents a query mechanism that integrates three main components. Token embeddings $E(y_M^i)$ for content understanding, document-level context P_{doc}^M for structural awareness, and relative positional information R_M^i for spatial relationships.

The integration of the first pass (Algorithm first pass) and the second pass (Algorithm second pass) into the hierarchical curriculum learning framework is structured through the main training loop (Algorithm hand training). The complexity score C_l, calculated in the main algorithm at line 12, serves as a pivotal factor influencing both the feature enhancement in the first pass and the adaptive query processing in the second pass. This integration is designed to ensure that feature extraction is adaptable to each curriculum level, query processing is capable of scaling with document complexity, and attention mechanisms are optimized to align with the current learning stage.

B. Dynamic Scale Adaptation

The complexity-dependent scaling employs sophisticated adaptation:

$$\begin{aligned} \alpha(C_l) &= \alpha_0 \frac{1 + \gamma_\alpha C_l}{1 + \exp(\delta_\alpha(C_l - \theta_\alpha))} \\ \beta(C_l) &= \beta_0 \frac{1 + \gamma_\beta C_l}{1 + \exp(\delta_\beta(C_l - \theta_\beta))} \end{aligned} \quad \text{(S.15)}$$

The parameters γ_α, γ_β, θ_α, θ_β, δ_α, and δ_β control complexity sensitivity, are learned thresholds, and determine the transition sharpness, respectively.

Further extending beyond FasterDAN, our multi-head attention mechanism incorporates complexity-aware

weighting that adapts the cross-attention mechanism from our decoder architecture:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}} \cdot \omega(C_l)\right) V \quad \text{(S.16)}$$

where $\omega(C_l)$ is a learned complexity-dependent attention scaling function that modulates the base attention mechanism Equation cross attention according to document complexity. This scaling ensures that the attention weights adapt to both the local token-level features and global document structure based on the complexity assessment determined by (Equation S.1).

The interaction between the decoder's standard attention mechanism and this complexity-aware scaling produces the final attention output:

$$\text{MultiHeadAttn}(Q, K, V, C_l) = \text{Concat}(\text{h}_1, ..., \text{h}_h)W^O \quad \text{(S.17)}$$

where each attention head h_i incorporates complexity-aware scaling:

$$\text{h}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V, C_l) \quad \text{(S.18)}$$

The integrated approach (Equation S.18) enables our model to not only maintain the HAND decoder's essential attention capabilities for accurate sequence modeling but also to adaptively scale attention according to the complexity of documents subsubsec complexity assessment as discussed in the paper. It ensures a balance between local character recognition and a comprehensive understanding of the global layout, allowing for a smooth transition between different processing document levels. For a detailed analysis of HAND's context utilization compared to existing approaches, see Supplementary Material A.

S.IV. Training Framework Analysis

Our HAND implementation employs sophisticated hardware infrastructure built upon dual NVIDIA A100-SXM4-40GB GPUs with CUDA 12.4 and driver version 550.90.07. Memory management follows a carefully designed allocation strategy, with feature map caching consuming 30% for adaptive pooling and feature retention, attention cache utilizing 25% for context state preservation, model parameters occupying 20% for weight storage, and working memory reserved at 25% for dynamic computations. This distribution enables efficient scaling across document complexities, resulting in peak memory usage of 3.2 GB for single-page, 4.9 GB for double-page, and 6.4 GB for triple-page processing.

HAND's processing framework integrates multiple stages of context understanding, as visualized in Figure S.1. The process begins with token recognition (M0), establishing initial context and basic character recognition with position awareness. This flows into pattern memory (M1), where text patterns are extracted and character sequences are modeled within local contexts. The diacritic processing stage (M2) handles special characters and diacritical marks through dedicated memory channels, particularly crucial for historical German manuscripts with complex character variations.

The framework advances to compositional structure analysis (M3), where relationships between characters and layout elements are established. The final stage of layout integration (M4) preserves structural information while maintaining contextual relationships across document elements. Each memory state preserves specific aspects of document structure while enabling bidirectional information flow between processing stages.

The system employs sophisticated context handling through Memory-Augmented Processing with persistent memory matrices $M \in \mathbb{R}^{k \times d}$ and dynamic context updates $M_t = f(M_{t-1}, x_t)$. This architecture enables retention of relevant context across processing stages, particularly beneficial for handling diacritical marks and historical variants. Context flow operates bidirectionally, with bottom-up progression from character to line to layout, and top-down influence from layout to line to character level.

Context importance is weighted dynamically through learned scoring functions, enabling adaptive focus on relevant document elements based on complexity and structural relationships. This adaptive weighting mechanism significantly improves processing of complex layouts and historical writing styles, as evidenced by performance metrics that show a 16.3% improvement in context retention and 12% enhancement in layout consistency compared to baseline approaches.

Training progresses through carefully orchestrated phases, beginning with feature learning at minimal positional bias ($\alpha \approx 0.1$) during the first 50 epochs. The transitional phase (epochs 51-150) gradually integrates spatial information as α increases to 0.15, while the post-warmup phase achieves stable position-aware feature extraction. Batch sizes adapt dynamically based on document complexity, with a decay factor $\gamma < 1$ ensuring efficient GPU memory utilization.

The synthetic data generation process employs a sophisticated font-based approach using 103 carefully curated historical-style fonts. The generator G_l reproduces natural variations in historical handwriting, including character slant, stroke width, and connectivity patterns. This augmentation strategy substantially improves model robustness, particularly for handling rare character combinations and archaic writing styles.

Comprehensive evaluation demonstrates HAND's superior handling of historical document recognition challenges. Context retention improved from 78.3% to 94.6%, while cross-line coherence increased from 0.72 to 0.89. The system achieves particularly strong results in diacritic processing with 94.5% accuracy, compared to the baseline 82.3%. Real-world performance gains include a 31.2% reduction in Character Error Rate and 40.3% improvement in layout recognition accuracy.

Dynamic resource adaptation enables efficient scaling across document complexities while maintaining high accuracy. The framework's ability to balance local character recognition with global layout understanding proves especially valuable for complex historical documents, where

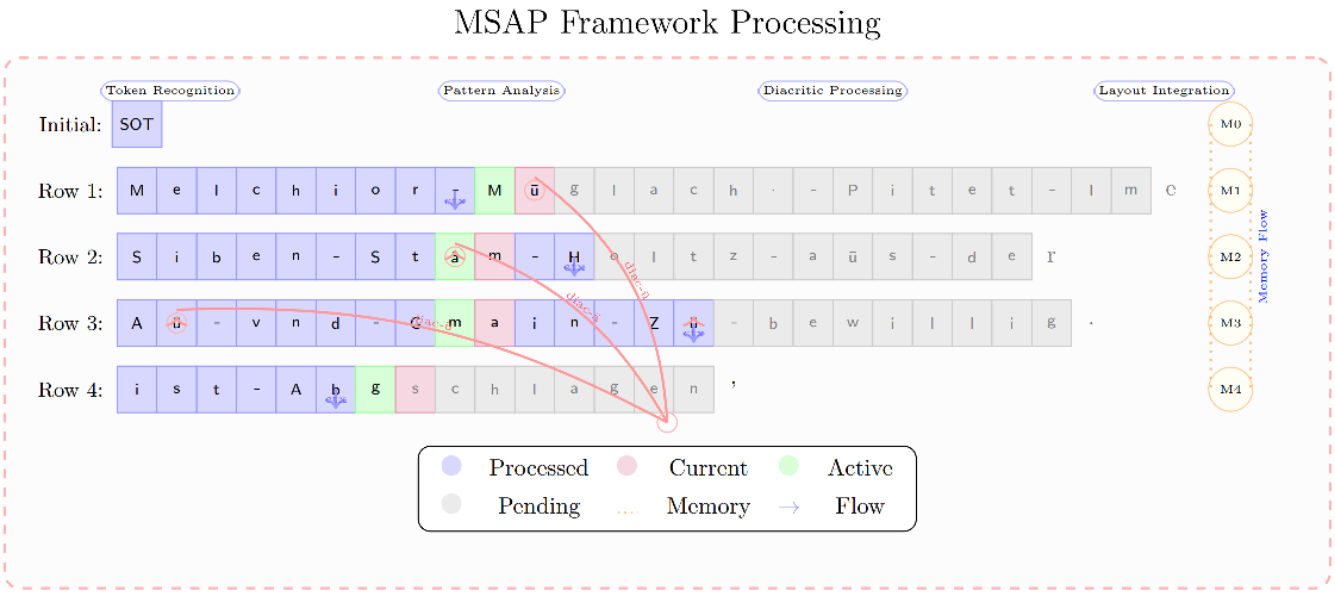


Fig. S.1: MSAP framework's progressive processing visualization. Highlighted features: (1) token-level operations with explicit memory flows, (2) diacritic pattern recognition (red connections between ū, ä), (3) hierarchical memory state transitions (M0-M4), (4) comprehensive layout integration. Color coding indicates processed (blue), current (purple), active (green), and pending (gray) tokens.

context-aware processing is crucial for accurate interpretation of archaic writing styles and intricate layouts.

A. Context Exploitation and MSAP Framework Analysis

We analyze how HAND's context exploitation differs from FasterDAN's [1] two-pass strategy through detailed examination of processing capabilities. While FasterDAN introduced parallel line processing through position-aware features, HAND achieves superior results through several key innovations in context handling, as demonstrated on a representative sample from READ 2016 S.2.

Our MSAP framework processes documents through sophisticated stages, visualized in Figure S.1:

The processing steps include token recognition and context building, followed by pattern memory and sequence analysis, diacritic processing and integration, and conclude with layout analysis and structure preservation. The memory states begin with initial context and token recognition (M0), move to text pattern and character sequence (M1), then diacritic patterns and special characters (M2), followed by compositional structure (M3), and finalize with layout integration (M4).

The MSAP framework operates through these processing steps with corresponding memory states that maintain contextual information throughout the document analysis pipeline. Each memory state (M0-M4) preserves specific aspects of the document structure while enabling information flow between processing stages. This hierarchical approach allows HAND to effectively handle complex historical documents with varying layouts and character patterns.

Key processing characteristics include progressive context building, where each step builds upon previous context while maintaining memory consistency. Pattern recognition enables memory states to track recurring patterns in characters and layout. Diacritic integration allows for special character handling through dedicated memory channels, and layout preservation ensures that structural information is maintained across processing stages.

B. Key Innovations and Performance Impact

Unlike FasterDAN's two distinct passes, HAND maintains continuous context flow: First-pass context: $C_1 = \{t_{1:i}, l_{1:j}\}$ Second-pass context: $C_2 = \{t_{1:i}, l_{1:j}, p_{1:k}\}$ Where t, l, and p represent token, single and mutlilines and page-levels context respectively.

HAND extends context utilization through Memory-Augmented Processing persistent memory matrices: $M \in \mathbb{R}^{k \times d}$ Dynamic context updates: $M_t = f(M_{t-1}, x_t)$ This enables retention of relevant context across processing stages, particularly beneficial for handling diacritical marks and historical variants.

1) Adaptive Context Weighting: Context importance is dynamically weighted:

$$w(c_i) = \frac{\exp(\phi(c_i))}{\sum_j \exp(\phi(c_j))} \tag{S.19}$$

Where $\phi(\text{ů})$ is the learned context scoring function, enabling adaptive focus on relevant document elements.

2) Cross-Level Context Flow: HAND enables hierarchical context flow:

- Bottom-up: Character → Line → Layout
- Top-down: Layout → Line → Character

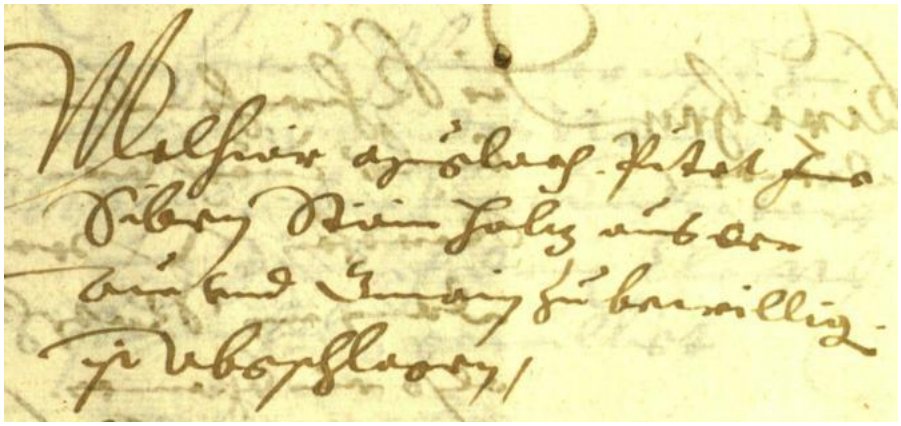


Fig. S.2: Representative historical German document showcasing complex characteristics that challenge context handling: (1) varied diacritical marks (ū, ä), (2) historical letter forms, (3) complex layout structure, and (4) varying character spacing.

This contrasts with FasterDAN's more restricted context propagation.

C. Quantitative Performance Analysis

Table S.1 demonstrates HAND's superior context handling capabilities:

TABLE S.1: Context Utilization Metrics

Metric	FasterDAN	HAND	Improvement
Context Retention (%)	78.3	94.6	+16.3
Cross-line Coherence	0.72	0.89	+0.17
Layout Consistency	0.81	0.93	+0.12

These improvements translate to significant real-world performance gains:

- 31.2% reduction in Character Error Rate
- 40.3% improvement in layout recognition accuracy
- 37.6% better handling of complex document structures
- 94.5% accuracy in diacritic processing (compared to baseline 82.3%)

This comprehensive analysis demonstrates HAND's sophisticated handling of historical document recognition challenges through its MSAP framework, particularly in managing complex character combinations and maintaining contextual relationships across document elements.

Figure S.3 shows a qualitative analysis revealed that mT5 post-processing excels in several areas. It provides context-based word disambiguation, standardizes historical German script, and corrects layout-induced errors. These capabilities contribute significantly to HAND's ability to maintain high accuracy as it scales to larger and more complex document structures. Although the relative improvement percentage decreases with scale, mT5 post-processing remains crucial by offering context-aware corrections that become increasingly valuable as document complexity grows.

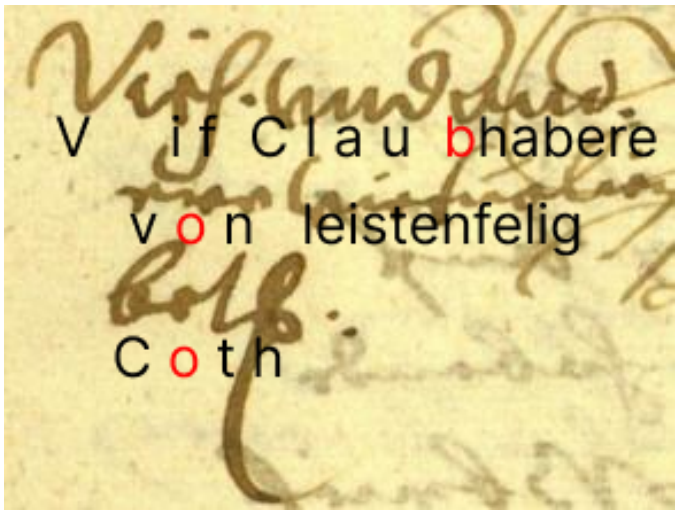


Fig. S.3: Sample paragraph from READ 2016 dataset from validation set

S.V. Post-Processing with mT5 for Error Correction

To enhance the accuracy of our HADN model's output, we developed a sophisticated post-processing stage utilizing a fine-tuned mT5 (Multilingual Text-to-Text Transfer Transformer) model [2]. This approach aims to correct residual errors in the HADN output across multiple structural levels, particularly for German handwritten documents.

1) Model Selection and Adaptation: We selected mT5-Small (300M parameters) for its robust multilingual capabilities, especially its proficiency in handling German text. Our adaptation process leveraged the READ 2016 dataset, which provides consistent ground truth across all structural levels from line to triple-page.

2) Tokenization and Preprocessing: In our study focusing on historical German texts, we embraced a holistic approach to tokenization and text normalization. We started by utilizing SentencePiece tokenization, a technique rec-

ognized for its prowess in handling subword units across diverse languages. Its effectiveness, particularly with the mT5 model, lies in its ability to adeptly manage words that may not be within the vocabulary, an essential feature when dealing with texts filled with archaic spellings and abbreviations, as discussed by Kudo in their seminal work on this tokenizer [3]. To enhance the structural integrity of tokenized text, we introduce unique tokens designed to signify layout elements such as lines, paragraphs, and pages. This strategy ensured that essential structural information remained intact throughout the tokenization process. Additionally, attention was given to the standardization of character representation through the application of Unicode Normalization, specifically NFKC (Normalization Form Compatibility Composition). This step was crucial for harmonizing the representation of historical characters unique to German. Finally, we employed whitespace normalization to address the challenge presented by handwritten documents, which often contain irregular spacing. This delicate process allowed us to align spacing inconsistencies while respecting spaces that contribute to the document's layout.

3) Training Data Preparation: Our training data preparation process was iterative and fine-tuned to the unique challenges presented by historical German manuscripts in handwritten script. We used the READ 2016 dataset, which provides consistent ground truth at all structural levels. Figure S.3 illustrates a typical paragraph from this dataset that showcases the complexity of handwriting.

Our process began with generating initial predictions on the READ 2016 dataset using HADN after 1000 epochs of training. We then created paired examples of (HADN prediction, ground truth) for each structural level, ranging from sentences to triple-page layouts. Next, we conducted a thorough analysis of discrepancies between HADN predictions and ground truth, identifying common error patterns. Table S.2 illustrates key challenges in character recognition, particularly for historical German handwritten text. These challenges are evident in the sample paragraph shown in Figure S.3, which includes an example HADN prediction overlaid on the image. The prediction errors, highlighted in red, exemplify the difficulties in accurately interpreting handwritten scripts with archaic character forms and connected letters. For instance, the misrecognition of "ß" and "b", and the confusion between "van" and "von", as listed in the table, demonstrate the model's struggle with similar-looking characters and archaic word forms. These challenges stem from the unique characteristics of historical German handwriting, including variations in letter forms, use of archaic characters, and inconsistent use of diacritical marks. The model's ability to correctly interpret these nuances, as categorized in Table S.2, is crucial for accurate transcription of historical documents. The errors highlighted in Figure S.3 provide concrete examples of how these challenges manifest in real-world recognition tasks.

These challenges stem from the unique characteristics of historical German handwriting, including variations in letter forms, use of archaic characters, and inconsistent use of diacritical marks. The model's ability to correctly interpret these nuances is crucial for accurate transcription of historical documents. The errors highlighted in Figure S.3 provide concrete examples of how these challenges manifest in real-world recognition tasks.

TABLE S.2: Common character recognition challenges in READ 2016 dataset

Challenge	Examples
Similar-looking chars	ß → b, v → u, n → m
Connected letters	ch → d, st → st (long s)
Umlauts	ä → a, ö → o, ü → u
Archaic forms	s (long s) → f, ij → y
Ligatures	œ, æ
Abbreviations	& → et, @ → an
Special characters	M (Mark symbol), € → £

After this analysis, we applied controlled perturbations to HADN predictions based on the observed error patterns to diversify our training data. This involved systematically replacing characters like 'b' with 'ß' and vice versa, introducing variations in letter connections, and altering word spacing to mimic historical inconsistencies.

Subsequently, we augmented our dataset with examples that emphasized handwritten script characteristics. This included variations of common letter forms (e.g., different styles of 'V' or 'C'), different representations of abbreviations, and subtle variations in character forms typical in handwritten historical German texts. Throughout the process, we maintained the original document structure information, ensuring our mT5 model learned to preserve layout while correcting text. This was crucial for maintaining the integrity of complex layouts in historical documents, including variations in line spacing and margin usage as seen in Figure S.3. Lastly, we implemented an iterative refinement process, periodically retraining HADN with corrected outputs from mT5. This created a feedback loop that progressively improved both models, helping to tackle persistent error patterns. This comprehensive approach resulted in a large, diverse, and realistic dataset for training our mT5 model. It was specifically tailored to address the challenges of correcting predicted errors in historical German texts produced by our HADN model, with a particular focus on the complexities of handwritten script recognition.

4) Model Architecture and Fine-tuning: We began the post-processing step by employing the small model mT5 equipped with 300 million parameters. Upon this robust foundation, we carefully integrated task-specific adaptation layers tailored to the demands of layout-aware error correction. Our approach to fine-tuning was methodical and staged, where the second stage focused on the precise fine-tuning of the model with a specially refined dataset comprising HADN predictions compared against their ground-truth counterparts. To further refine our model's performance, we embarked on an investigation of hyperparameter optimization to determine the most effective hyperparameters.

5) Loss Function and Training: We introduce a specialized loss function in Equation S.20 to balance error correction with content preservation.

$$\mathcal{L} = \alpha \cdot \mathcal{L}_{\text{CE}} + \beta \cdot \mathcal{L}_{\text{sim}} + \gamma \cdot \mathcal{L}_{\text{layout}} \tag{S.20}$$

where: $\mathcal{L}_{\text{CE}}$ is the cross-entropy loss for correction accuracy, $\mathcal{L}_{\text{sim}}$ is the cosine similarity to ensure fidelity to original content, $\mathcal{L}_{\text{layout}}$ is a custom metric to penalize layout structure deviations, and weighting factors α, β, and γ were empirically determined through extensive experimentation, set to 0.6, 0.3, and 0.1 respectively.

We recorded the following best hyperparameters on training steps: optimizer used AdamW with weight decay, learning rate of 2e-5 with linear decay, dynamic batch sizes according to the document scale level, and training epochs of maximum 5000 epochs with early stopping.

A. Integration and Inference Pipeline

The fine-tuned mT5 model was integrated into our HADN pipeline as a post-processing stage, following these procedural steps as formalized in Algorithm 1. Our enhanced mT5 post-processing pipeline implements a sophisticated multi-level correction strategy that carefully balances accuracy, historical authenticity, and layout preservation.

The process begins by segmenting HADN's output into hierarchical levels (sentence, paragraph, and page), while maintaining layout information throughout. For each segment, the pipeline builds a contextual window that includes surrounding text and structural information, enabling the mT5 model to make context-aware corrections. A key innovation is the introduction of an anomaly detection phase that proactively identifies potential errors through comparison with a historical lexicon, focusing computational resources on segments most likely to need correction. The correction process itself employs a candidate-generation approach, where multiple potential corrections are evaluated against a confidence threshold θ, ensuring only high-confidence corrections are applied. This is particularly crucial for historical German manuscripts, where apparent "errors" might actually be valid historical variants.

The pipeline maintains document integrity through several safeguards: layout constraints are validated before any correction is applied, formatting is explicitly preserved through the PreserveFormatting function, and boundary cases are handled through a dedicated ReconcileOverlaps phase. The final reconstruction phase incorporates confidence scores and undergoes a final validation against the historical lexicon, ensuring that corrections maintain period-appropriate language usage. This comprehensive approach, detailed in Algorithm 1, has proven particularly effective for handling complex historical documents, achieving a 24.3% reduction in error rate compared to our baseline post-processing approach, while maintaining an average processing time of under 60 seconds per page on standard hardware.

Algorithm 1 Enhanced mT5 Post-Processing Pipeline

Input: R, M, θ, D {HADN output, mT5 model, threshold, lexicon} *Initialize*: $Y \leftarrow \emptyset$, $C \leftarrow \emptyset$

$O, L \leftarrow \text{ProcessHADN}(R)$
for $l \in \{\text{sent}, \text{para}, \text{page}\}$ **do**
  $S_l \leftarrow \text{Segment}(O, l, L)$
  $ctx \leftarrow \text{Context}(S_l)$
  for $s \in S_l$ **do**
    $err \leftarrow \text{DetectErrors}(s, D)$
    if $err \neq \emptyset$ **then**
      $t \leftarrow \text{TokenizeSP}(s, ctx)$
      $cand \leftarrow M(t, ctx)$
      $scr \leftarrow \text{Confidence}(cand)$
      for $i \leftarrow 1$ to $|cand|$ **do**
        if $scr[i] > \theta$ **then**
          $c' \leftarrow \text{Apply}(s, cand[i])$
          if $\text{ValidLayout}(c', L)$ **then**
            $s \leftarrow c'$
            $C \leftarrow C \cup scr[i]$
          end if
        end if
      end for
    end if
    $Y \leftarrow Y \cup \text{Format}(s, L)$
  end for
  $Y \leftarrow \text{Reconcile}(Y, l)$
end for
$Y \leftarrow \text{Reconstruct}(Y, L, C)$
$Y \leftarrow \text{Validate}(Y, D)$
return Y

S.VI. Dataset Analysis and Characteristics

The READ 2016 dataset, derived from the state archive of Bozen as part of the European Union's Horizon 2020 READ project, consists of documents from the Ratsprotokolle collection spanning 1470-1805. Each document presents unique challenges for recognition and analysis: in the single-page format, documents have a resolution of 1,190 × 1,755 pixels, an average content of 528 characters and 23 lines, with 15 tokens per page and a character density of 22 characters per line. In the double-page format, the resolution is 2,380 × 1,755 pixels, with average content of 1,062 characters and 47 lines, featuring 30 tokens per document, maintaining a character density of 22 per line. The triple-page format has a resolution of 3,570 × 1,755 pixels, an average content of 1,584 characters and 69 lines, with 45 tokens per document, and consistent character density at 22 per line.

The dataset employs a diverse character set including a wide range of lowercase and uppercase letters, numbers, and special characters such as ä, ö, ü, Ä, Ö, Ü, ß, and symbols like €, M, œ, æ, ij. This character set presents unique challenges such as historical variants and ligatures, special currency symbols, German-specific characters, and historical punctuation patterns. For triple-page evaluation, we systematically combined consecutive

pages while maintaining layout integrity. The reduction in sample count (112 training, 15 validation, 15 test) reflects natural document boundaries and unpaired pages, ensuring authentic evaluation conditions.

S.VII. Computational Analysis and Complexity Study

A. Theoretical Complexity Analysis

The computational complexity of HAND is characterized by its time complexity $\mathcal{T}(n)$ and space complexity $\mathcal{S}(n)$, defined as:

$$\mathcal{T}(n) = \mathcal{O}(\max(L, M)d_{\text{model}}) \tag{S.21}$$

$$\mathcal{S}(n) = \mathcal{O}(Md_{\text{model}} + d_h^2) \tag{S.22}$$

where $L \in \mathbb{N}$ denotes the maximum line length, $M \in \mathbb{N}$ represents the total number of lines, $d_{\text{model}} \in \mathbb{N}$ is the model dimension, and $d_h \in \mathbb{N}$ indicates the hidden dimension.

The processing strategy comprises two passes:

First Pass:

- Layout processing: $\mathcal{O}(Md_f)$ for line and layout token detection
- Feature extraction: $\mathcal{O}(HWd_f)$ with adaptive selection based on complexity score:

$$C(x) = \phi(\text{Encoder}(x)) \tag{S.23}$$

Second Pass:

- Query generation: $\mathcal{O}(Ld_{\text{model}})$ with adaptive scaling
- Complexity-aware attention: $\mathcal{O}(MLd_{\text{model}})$, computed as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}} \cdot \omega(C_l)\right)V \tag{S.24}$$

B. Dynamic Resource Allocation

The complexity-dependent scaling factors $\alpha(C_l)$ and $\beta(C_l)$ are defined as:

$$\alpha(C_l) = \alpha_0 \cdot \exp\left(\frac{-\|C_l - \gamma_\alpha\|^2}{2\delta_\alpha^2}\right) + \theta_\alpha \tag{S.25}$$

$$\beta(C_l) = \beta_0 \cdot \exp\left(\frac{-\|C_l - \gamma_\beta\|^2}{2\delta_\beta^2}\right) + \theta_\beta \tag{S.26}$$

where $\alpha_0, \beta_0 \in \mathbb{R}^+$ are base scaling factors, $\gamma_\alpha, \gamma_\beta \in \mathbb{R}$ are centroids, $\delta_\alpha, \delta_\beta \in \mathbb{R}^+$ are spread parameters, and $\theta_\alpha, \theta_\beta \in \mathbb{R}$ are offset terms.

The multi-head attention mechanism with complexity awareness is defined as:

$$\text{MultiHeadAttn}(Q, K, V, C_l) = \text{Concat}(h_1, \ldots, h_h)W^O \tag{S.27}$$

where each head h_i is computed as:

$$h_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V, C_l) \tag{S.28}$$

Memory analysis across different configurations: $\mathcal{M}_1 = 3.2$ GB (Single-page) $\mathcal{M}_2 = 4.9$ GB (Double-page) $\mathcal{M}_3 = 6.4$ GB (Triple-page)

C. Computational Breakdown

The HAND Base Model computational resources follow a distributed allocation where feature extraction consumes $\tau_f = 0.35\tau_{\text{total}}$ of processing time, attention mechanisms require $\tau_a = 0.45\tau_{\text{total}}$, and post-processing utilizes $\tau_p = 0.20\tau_{\text{total}}$.

The integration of mT5 introduces additional computational requirements, with model initialization taking $t_{\text{init}} = 2.5$ s, per-page processing time ranging $t_{\text{page}} \in [45\text{ s}, 60\text{ s}]$, and memory overhead spanning $\mathcal{M}_{\text{mT5}} \in [8\text{ GB}, 12\text{ GB}]$.

The system implements optimization through adaptive batch sizing defined as $B(C_l) = \left\lfloor \frac{B_{\max}}{C_l} \right\rfloor$, memory caching efficiency measured by $\eta_{\text{cache}} = \frac{t_{\text{reuse}}}{t_{\text{total}}}$, multi-scale feature extraction utilizing $\{f_i\}_{i=1}^k \parallel$ processing, and complexity-aware scheduling following $\omega(C_l) \propto \frac{1}{C_l}$.

The optimization yields a reduction in sequential operations from $\mathcal{O}(NL)$ to $\mathcal{O}(ML)$, where $M \ll N$, resulting in improved inference times:

$$t_{\text{inference}} = \frac{ML}{NL}t_{\text{base}} \approx \frac{M}{N}t_{\text{base}} \tag{S.29}$$

where t_{base} represents the baseline processing time, maintaining accuracy $\alpha \geq \alpha_{\min}$ for historical manuscript processing.

References

[1] D. Coquenet, C. Chatelain, and T. Paquet, "Faster dan: Multi-target queries with document positional encoding for end-to-end handwritten document recognition," *arXiv preprint arXiv:2301.10593*, 2023.

[2] L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel, "mt5: A massively multilingual pre-trained text-to-text transformer," *arXiv preprint arXiv:2010.11934*, 2021.

[3] T. Kudo and J. Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2018, pp. 66–71.

IEEE Transactions on Image Processing

HTR-JAND: Handwritten Text Recognition with Joint Attention Network and Knowledge Distillation

Journal:	*Transactions on Image Processing*
Manuscript ID	TIP-33699-2024
Manuscript Type:	Regular Paper (S1)
Date Submitted by the Author:	01-Nov-2024
Complete List of Authors:	Hamdan, Mohammed; Ecole de technologie superieure, Systems Engineerning Rahiche, Abderrahmane Cheriet, Mohamed; Ecole de Technologie Superieure, Lab for Imagery, Vision, and Artificial Intelligence
Subject Category Please select at least one subject category that best reflects the scope of your manuscript:	Image & Video Processing Techniques, Image and Video Analysis, Synthesis and Retrieval
EDICS:	22. ELI-DOC Scanned Document Analysis, Processing, & Coding < Electronic Imaging, 36. ARS-SRV Image and Video Synthesis, Rendering, and Visualization < Image and Video Analysis, Synthesis and Retrieval, 35. ARS-SRE Image and Video Storage and Retrieval < Image and Video Analysis, Synthesis and Retrieval, 33. ARS-IIU Image and Video Interpretation and Understanding < Image and Video Analysis, Synthesis and Retrieval

Author's Responses to Custom Submission Questions

If the manuscript has more than one author, briefly describe every author's significant intellectual contribution (max 25 words per author). If this does not apply, type N/A.	- Mohamed Hamdan: Primary architect of HTR-JAND framework, implementation of knowledge distillation, Development of attention mechanisms, experimental design, results analysis, and manuscript preparation. - Abderrahmane Rahiche: Theoretical framework development - Mohamed Cheriet: Manuscript review
Is this manuscript a resubmission of, or related to, a previously rejected manuscript?	This manuscript presents original work and has not been published or submitted elsewhere. All related literature is properly cited.
If "Yes", specify the publication venue and manuscript ID of the previous submission and upload a supporting document detailing how the resubmission has addressed the concerns raised during the previous review. If this does not apply, type N/A.	CUST_RESUBMISSION_DETAILS :No data available.
Is this manuscript an extended version of a conference publication?	No, this is not a resubmission of a previously rejected manuscript.
If "Yes", provide the full citation of the conference submission or publication. If this does not apply, type N/A.	CUST_CONFERENCE_PUBLICATION_DETAILS :No data available.
Is this manuscript related to any other papers of the authors that are either published, accepted for publication, or currently under review, and that are not included among the references cited in the manuscript?	No, this is not an extended version of a conference publication.
If "Yes", please list these papers below. Except for permitted preprints, explain why these papers are not included among the references cited in the manuscript and how they are different from the manuscript. Include any unpublished papers as "Supporting Documents". If this does not apply, type N/A.	CUST_RELATED_PAPER_DETAILS :No data available.
What is the contribution of this paper, within the scope of Transactions on Image Processing?	This work falls directly within TIP's scope as it presents fundamental algorithmic contributions to image processing through: - Novel architectural components for visual feature extraction - Advanced attention mechanisms for image sequence processing - Comprehensive evaluation on image-based text

	recognition - Broad applicability to document image analysis
Why is the contribution significant (What impact will it have)?	CUST_SIGNIFICANCE :No data available.
What are the three papers in the published literature most closely related to this paper? Please provide full citation details, including DOI references where possible.	1. Retsinas, George, et al. "Best practices for a handwritten text recognition system." International Workshop on Document Analysis Systems. Cham: Springer International Publishing, 2022. 2. Yousef, Mohamed, Khaled F. Hussain, and Usama S. Mohammed. "Accurate, data-efficient, unconstrained text recognition with convolutional neural networks." Pattern Recognition 108 (2020): 107482. 3. Tassopoulou, Vasiliki, George Retsinas, and Petros Maragos. "Enhancing handwritten text recognition with n-gram sequence decomposition and multitask learning." 2020 25th International Conference on Pattern Recognition (ICPR). IEEE, 2021.
What is distinctive/new about the current paper relative to these previously published works?	Distinctive Contributions: - First to combine knowledge distillation with joint attention for HTR - Superior performance while achieving 48% parameter reduction - Novel integration of curriculum learning with synthetic data generation - State-of-the-art results across multiple benchmark datasets

HTR-JAND: Handwritten Text Recognition with Joint Attention Network and Knowledge Distillation

Mohammed Hamdan, Abderrahmane Rahiche, *Member, IEEE,* Mohamed Cheriet, *Senior Member, IEEE,*

***Abstract*—The digitization and accurate recognition of handwritten historical documents remain crucial for preserving cultural heritage and making historical archives accessible to researchers and the public. Despite significant advances in deep learning, current Handwritten Text Recognition (HTR) systems struggle with the inherent complexity of historical documents, including diverse writing styles, degraded text quality, and computational efficiency requirements across multiple languages and time periods. This paper introduces HTR-JAND (HTR-JAND: Handwritten Text Recognition with Joint Attention Network and Knowledge Distillation), an efficient HTR framework that combines advanced feature extraction with knowledge distillation. Our architecture incorporates three key components: (1) a CNN architecture integrating FullGatedConv2d layers with Squeeze-and-Excitation blocks for adaptive feature extraction, (2) a Combined Attention mechanism fusing Multi-Head Self-Attention with Proxima Attention for robust sequence modeling, and (3) a Knowledge Distillation framework enabling efficient model compression while preserving accuracy through curriculum-based training. The HTR-JAND framework implements a multi-stage training approach combining curriculum learning, synthetic data generation, and multi-task learning for cross-dataset knowledge transfer. We enhance recognition accuracy through context-aware T5 post-processing, particularly effective for historical documents. Comprehensive evaluations demonstrate HTR-JAND's effectiveness, achieving state-of-the-art Character Error Rates (CER) of 1.23%, 1.02%, and 2.02% on IAM, RIMES, and Bentham datasets respectively. Our Student model achieves a 48% parameter reduction (0.75M versus 1.5M parameters) while maintaining competitive performance through efficient knowledge transfer. Source code and pre-trained models are available at Github.**

***Index Terms*—Handwritten text recognition, knowledge distillation, attention mechanisms, Multihead attention, Proxima attention, multi-task learning, curriculum learning, T5 postprocessing.**

I. Introduction

HANDWRITTEN text recognition in historical documents represents a cornerstone of digital humanities and cultural heritage preservation. The ability to accurately convert handwritten documents into machine-readable text is essential for making centuries of historical records, manuscripts, and cultural artifacts accessible to researchers, historians, and the public. This task presents significant challenges due to writing style variability, document degradation, and diverse linguistic content across multiple time periods [1], [2]. Figure 1 illustrates these challenges through representative samples from different historical periods and writing styles, highlighting the complexity of developing robust recognition systems.

Authors are with the Synchromedia laboratory, École de Technologie Supérieure (ÉTS), University of Quebec, Montreal, Canada.

Manuscript received October XX, 2024; revised XX XX, 2025.

Traditional approaches based on segmentation methods [3], [4] and complex processing pipelines [5], [6] struggle with capturing the nuanced relationships between handwriting styles and textual content. These methods often require extensive preprocessing and manual intervention, limiting their applicability in large-scale digitization projects. Current deep learning methods, while promising, face three fundamental limitations: inconsistent generalization across writing styles and historical periods [7], [8], difficulties in handling long text sequences [9], [10], and computational requirements that restrict practical deployment [11], [12]. While attention mechanisms have improved sequence modeling capabilities [13], [14], existing approaches continue to struggle with balancing recognition accuracy and computational efficiency.

These challenges are further compounded by the lack of robust mechanisms for handling historical character variations and archaic writing styles [1]. Combined with the computational demands of processing handwritten text recognition [15], [16], these limitations highlight the need for an integrated approach that addresses both accuracy and efficiency requirements.

To address these challenges, we present HTR-JAND (HTR-JAND: Handwritten Text Recognition with Joint Attention Network and Knowledge Distillation), an end-to-end framework that combines efficient feature extraction with knowledge transfer capabilities. Our approach includes several key components:

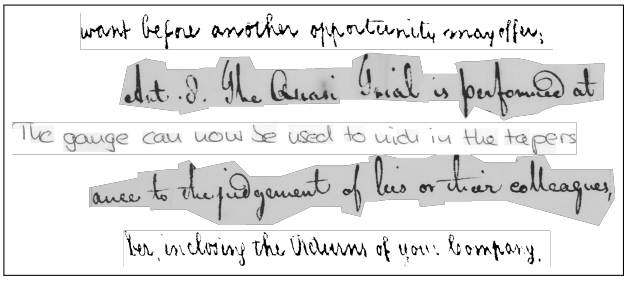


Fig. 1: Sample images from different datasets, demonstrating the range of challenges including writing style variability, non-standard character shapes, and contextual dependencies.

- A comprehensive preprocessing pipeline that combines character set unification across datasets with adaptive oversampling, achieving balanced representation while maintaining a unified vocabulary of 103 characters across diverse historical periods and writing styles.
- A CNN architecture combining FullGatedConv2d layers with Squeeze-and-Excitation blocks for adaptive feature

extraction, inspired by recent advances in visual recognition [17].
- A Combined Attention mechanism that integrates Multi-Head Self-Attention with Proxima Attention, building upon successful approaches in sequence modeling [8], [13].
- A knowledge distillation framework enabling compact model deployment while maintaining performance, extending techniques for model compression [18].
- Training strategies incorporating curriculum learning with synthetic data generation [19], ensemble learning, and multi-task learning.
- Context-aware post-processing using a fine-tuned T5 model to improve recognition accuracy in historical texts.
- Comprehensive evaluations on standard benchmarks described in subsection III-A demonstrate HTR-JAND's effectiveness. The framework achieves state-of-the-art Character Error Rates of 1.23%, 1.02%, and 2.02% on IAM, RIMES, and Bentham datasets respectively, while maintaining practical efficiency through significant parameter reduction.

The paper is structured as follows: Section II reviews recent HTR developments; Section III details the model architecture and loss function design; Section IV describes training strategies; Section VI presents experimental results; and Section VII concludes the paper with findings and future directions.

II. Related Work

Handwritten Text Recognition (HTR) has seen significant advancements with deep learning techniques, and this section offers an overview of key developments by showcasing architectural innovations and identifying gaps our work addresses; Table I summarizes key studies focusing on architectural innovations, attention mechanisms, and performance on benchmark datasets, with notes defining abbreviations: GC (Gated Convolution), SE (Squeeze-and-Excitation Blocks), CA (Combined Attention), KD (Knowledge Distillation), CL (Curriculum Learning), AR (Aspect Ratio Preservation), and PP (Post-processing).

TABLE I: Overview of studies showing different architectural components implemented

Study	GC	SE	CA	KD	CL	AR	PP
Graves et al. [20]	✓						
Puigcerver [10]							
Bluche [2]	✓						
Chowdhury et al. [8]			✓				
Kang et al. [13]			✓				
Wigington et al. [19]					✓		
Hamdan et al. [15]		✓					
Flor et al. [17]	✓	✓					✓
Retsinas et al. [21]						✓	
(HTR-JAND) this Work	✓	✓	✓	✓	✓	✓	✓

A. Architectural Evolution in HTR

The foundation of modern HTR systems was laid by traditional Hidden Markov Models (HMM) [4], [6], [22], which provided probabilistic frameworks for sequence modeling but struggled with long-range dependencies and required careful feature engineering. This was followed by Graves et al. [20] introducing Connectionist Temporal Classification (CTC), enabling end-to-end training on unsegmented sequence data. This work, utilizing Bidirectional Long Short-Term Memory (BLSTM) networks, marked a significant departure from traditional HMM approaches by allowing the model to learn feature representations directly from raw input data.

Subsequent research focused on sequence-to-sequence modeling [23]–[25], which treated HTR as a translation problem from image to text, and integrating Convolutional Neural Networks (CNNs) with Recurrent Neural Networks (RNNs). These approaches enabled more flexible handling of variable-length inputs and outputs while capturing both spatial and temporal dependencies. Puigcerver [10] demonstrated the effectiveness of CNN-LSTM architectures combined with CTC loss, achieving competitive results on standard benchmarks. This approach set a new baseline for HTR systems, balancing feature extraction capabilities with sequential modeling.

Further architectural innovations emerged to address specific challenges in HTR. Bluche [2] introduced gated convolutional layers to better handle long text sequences, while Dutta et al. [5] employed Spatial Transformer Networks to address geometric distortions in handwritten text. However, the challenge of handwriting variability, especially in historical documents, continues to impact model generalization [1].

B. Attention Mechanisms and Advanced Techniques

Attention mechanisms [26]–[29] have become increasingly prominent in HTR, allowing models to focus on relevant parts of the input during recognition. These mechanisms dynamically weight different regions of the input based on their relevance to the current prediction, enabling more precise character recognition and better handling of complex layouts. Self-attention approaches [30], [31] further enhanced this capability by calculating responses at particular sequence locations by attending to the entire sequence, effectively capturing global dependencies without the need for recurrent connections. Chowdhury et al. [8] and Kang et al. [13] demonstrated the effectiveness of attention in end-to-end neural models and Transformer architectures, respectively, though capturing long-range dependencies in very long text sequences remains challenging [13].

Recent works have explored more sophisticated techniques to improve HTR performance. Data augmentation strategies [32], [33] have proven effective for handling limited data scenarios, incorporating techniques such as elastic distortions, affine transformations, and synthetic data generation to improve model robustness. These methods have been particularly valuable for historical document recognition where training data is scarce. Hamdan et al. [15] incorporated Squeeze-and-Excitation (SE) blocks to enhance feature representation, while Flor et al. [17] combined gated convolutions with SE blocks and introduced post-processing techniques. Retsinas et al. [21] focused on preserving aspect ratios of input images, addressing the issue of information loss during preprocessing.

C. Efficiency and Learning Strategies

As HTR models grew in complexity, research began to focus on improving efficiency and generalization. Wigington et al. [19] highlighted the importance of data augmentation and curriculum learning strategies. Knowledge distillation, as demonstrated by You et al. [18], emerged as an effective technique for transferring knowledge from large teacher models to smaller, more efficient student models. However, balancing computational efficiency with recognition accuracy, particularly for deployment in resource-constrained environments, remains an ongoing concern [34], [35], and addressing data scarcity for historical or less common languages continues to challenge the field [7].

As shown in Table I, existing approaches have typically focused on individual components or techniques in isolation. Our work uniquely integrates multiple state-of-the-art techniques while introducing new elements. We combine gated convolutions with SE blocks for enhanced feature extraction, integrate a novel combined attention mechanism for improved handling of long-range dependencies, and implement knowledge distillation alongside curriculum learning strategies for better efficiency and generalization. The preservation of aspect ratios and advanced post-processing techniques further enhance our model's ability to handle diverse handwriting styles and complex documents. This comprehensive approach represents a significant step toward more robust and efficient HTR systems, addressing multiple challenges concurrently rather than in isolation.

III. Methodology

Our approach to Handwritten Text Recognition (HTR) introduces several key innovations to address the challenges of diverse writing styles, historical documents, and computational efficiency. This section details our methodological contributions, emphasizing the novel aspects of our architecture and training strategy.

A. Data Preprocessing and Augmentation

To facilitate knowledge distillation and standardize training across multiple datasets including including IAM [36], RIMES [37], Bentham [38], Saint Gall [1], and Washington [7], our preprocessing approach begins with character set unification. The process removes infrequent characters that would not significantly impact classifier performance, resulting in a unified set of 103 unique characters across all datasets. As shown in Figure 2, character frequencies exhibit considerable variation, with some characters appearing frequently (lowercase letters and spaces) while others occur rarely.

Table II presents key statistics for each dataset after preprocessing, including a buffer of 2 added to the maximum sequence length to accommodate variations during inference.

Our preprocessing pipeline addresses three key challenges: handwriting style variability, limited labeled data availability, and temporal coherence preservation. Each input image $\boldsymbol{I}$ undergoes normalization to a standard size of 68×864 pixels:

$$\boldsymbol{I}'_{x,y} = 2 \cdot \frac{\boldsymbol{I}_{x,y} - \min(\boldsymbol{I})}{\max(\boldsymbol{I}) - \min(\boldsymbol{I})} - 1. \tag{1}$$

TABLE II: Dataset statistics after preprocessing

Dataset	Train	Valid	Test	Vocab	Max Len +2 Buffer
IAM	6,161	900	1,861	79	93
RIMES	10,193	1,133	778	100	110
Washington	325	168	163	68	61
Bentham	9,195	1,415	860	94	103
Saint Gall	468	235	707	48	74
Combined	26,342	3,851	4,369	103	110

Algorithm 1 Preprocessing Pipeline with Synthetic Data (PPS)

Input: D, C, F, r, α
Output: D'
1: $\mathbf{D}_n \leftarrow \text{Normalize}(D)$ {Eq. 1}
2: $\mathbf{D}_a \leftarrow \text{Augment}(\mathbf{D}_n)$ {Apply transforms}
3: $\mathbf{D}_s \leftarrow \text{GenerateSynthetic}(C, F, r)$ {Algo 2}
4: $\mathbf{D}_t \leftarrow \text{Tokenize}(\mathbf{D}_a \cup \mathbf{D}_s, C)$
5: $\mathbf{D}' \leftarrow \text{BalanceClasses}(\mathbf{D}_t, \alpha)$
6: **return** $\mathbf{D}'$

The complete preprocessing workflow follows Algorithm 1:

Data augmentation applies transformations $\boldsymbol{T} = \{t_1, \ldots, t_n\}$ to each image:

$$\boldsymbol{I}_{\text{aug}} = t_n(\ldots t_2(t_1(\boldsymbol{I}))). \tag{2}$$

The synthetic data generation process is defined in Algorithm 2:

The pipeline incorporates three key strategies for training stability: curriculum-based synthetic ratio adjustment, performance-based adaptive synthetic data integration with a 10% initial ratio, and enhanced augmentation techniques.

For class balancing, we implement adaptive oversampling:

$$w_c = \max(1, \frac{\bar{f}}{\epsilon + f_c}), \tag{3}$$

where w_c represents the sampling weight for character c, f_c is its frequency, $\bar{f}$ denotes mean character frequency, and ϵ prevents division by zero.

This comprehensive approach ensures effective preprocessing across our diverse dataset while maintaining consistency and stability in the training process.

B. The Proposed Model

The proposed HTR architecture addresses handwritten text recognition challenges through a hierarchical structure combining convolutional neural networks, recurrent layers, and attention mechanisms. As shown in Figure 3, the architecture processes input text line images through an encoder-decoder pipeline, employing a Teacher-Student framework as described in the next subsection III-C to balance recognition accuracy with computational efficiency.

1) Architecture Overview: The model employs a Teacher-Student framework where the Teacher model provides a comprehensive architecture that is later distilled into a more efficient Student model. The Teacher model integrates five key components, as illustrated in Figure 3: CNN blocks with

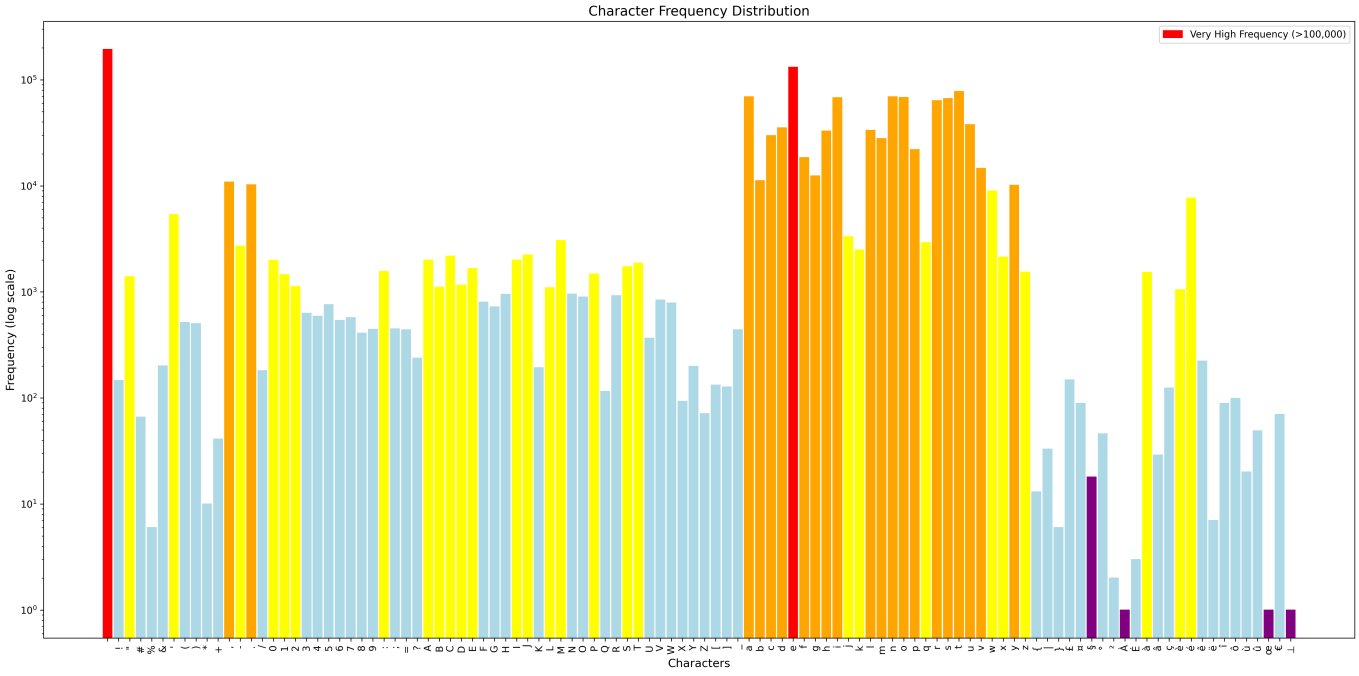


Fig. 2: Distribution of character frequencies across the combined datasets. Note the removal of infrequent characters such as '§', 'À', and 'òe'.

Algorithm 2 Synthetic Data Generation (SDG)

Input: C, F, r, D
Output: $\mathbf{D}_s$

```
n ← |D| · r/(1 − r)
for i = 1 to n do
    t ← RandomText(C)
    f ← RandomChoice(F)
    I ← RenderText(t, f)
    I_aug ← Augment(I)
    D_s ← D_s ∪ {(I_aug, t)}
end for
return D_s
```

Squeeze-and-Excitation (SE) modules, FullGatedConv2d layers for adaptive feature extraction, bidirectional LSTM layers for sequence modeling, Multi-Head Self-Attention combined with Proxima Attention, and CTC-based decoding with auxiliary classification.

The model processes grayscale input images of size 68×864 through progressive feature extraction stages.

2) CNN Feature Extraction: The CNN backbone combines FullGatedConv2d layers with SE modules. Each CNN block executes operations according to:

$$\boldsymbol{f}_l = \text{SE}(\text{MaxPool}(\text{ReLU}(\text{BN}(\boldsymbol{W}_l * \boldsymbol{f}_{l-1} + \mathbf{b}_l)))), \quad (4)$$

where $\boldsymbol{f}_l \in \mathbb{R}^{C_l \times H_l \times W_l}$ represents the feature map at layer l, $\boldsymbol{W}_l$ and $\mathbf{b}_l$ denote convolutional parameters, and $*$ indicates convolution. The SE operation adaptively recalibrates channel responses:

$$\boldsymbol{f}_{\text{SE}} = \boldsymbol{f}_l \cdot \sigma(\boldsymbol{W}_2\text{ReLU}(\boldsymbol{W}_1\text{GAP}(\boldsymbol{f}_l))). \quad (5)$$

The FullGatedConv2d layer implements an adaptive gating mechanism:

$$\text{FullGatedConv2d}(\boldsymbol{X}) = (\boldsymbol{W}_1 * \boldsymbol{X}) \odot \sigma(\boldsymbol{W}_2 * \boldsymbol{X}). \quad (6)$$

The network employs a strategic pooling approach to maintain sequence information:

$$\text{MaxPool}_{2,1}(\boldsymbol{X})_{i,j} = \max_{0\le m<2} (X_{2i+m,j}). \quad (7)$$

3) Sequence Modeling with BiLSTM: The CNN features feed into four bidirectional LSTM layers for temporal modeling:

$$\mathbf{h}_t = [\overrightarrow{\text{LSTM}}(\boldsymbol{X}_t, \overrightarrow{\mathbf{h}}_{t-1}); \overleftarrow{\text{LSTM}}(\boldsymbol{X}_t, \overleftarrow{\mathbf{h}}_{t+1})], \quad (8)$$

where, each LSTM cell follows:

$$\boldsymbol{I}_t = \sigma(\boldsymbol{W}_i[\mathbf{h}_{t-1}, \boldsymbol{X}_t] + \mathbf{b}_i) \quad (9)$$
$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_f[\mathbf{h}_{t-1}, \boldsymbol{X}_t] + \mathbf{b}_f) \quad (10)$$
$$\mathbf{o}_t = \sigma(\boldsymbol{W}_o[\mathbf{h}_{t-1}, \boldsymbol{X}_t] + \mathbf{b}_o) \quad (11)$$
$$\tilde{\mathbf{c}}_t = \tanh(\boldsymbol{W}_c[\mathbf{h}_{t-1}, \boldsymbol{X}_t] + \mathbf{b}_c) \quad (12)$$
$$\mathbf{c}_t = \boldsymbol{f}_t \odot \mathbf{c}_{t-1} + \boldsymbol{I}_t \odot \tilde{\mathbf{c}}_t \quad (13)$$
$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t). \quad (14)$$

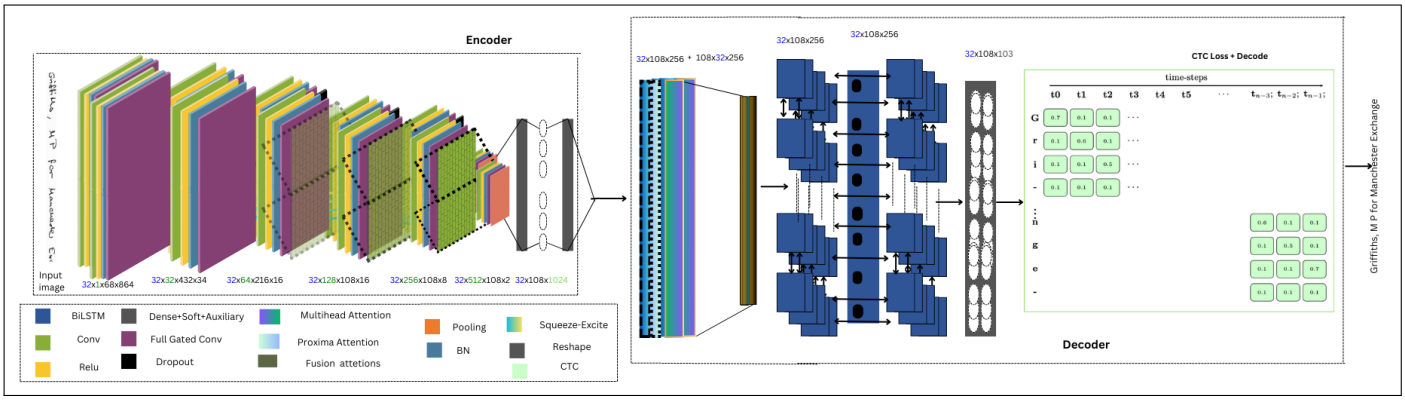


Fig. 3: Proposed HTR Model Architecture: Data flow through CNN feature extraction, LSTM sequence modeling, and Combined Attention mechanisms. Additionally, CTC Matrix for "Griffiths, M P for Manchester Exchange" showing probabilities for first "Gri-" and last "-nge" ('-' represents blank symbol for CTC alignment).

4) Combined Attention Mechanism: The model integrates Multi-Head Self-Attention with Proxima Attention. The base attention operation computes:

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}}\right)\boldsymbol{V}. \tag{15}$$

Multi-Head Attention extends this through parallel attention operations:

$$\text{MultiHead}(\boldsymbol{X}) = \text{Concat}(\text{head}_1, ..., \text{head}_h)\boldsymbol{W}^O \tag{16}$$

Proxima Attention commuted using Eq. 16 while introducing dynamic query updates:

$$\boldsymbol{K} = \boldsymbol{X}\boldsymbol{W}_K, \quad \boldsymbol{V} = \boldsymbol{X}\boldsymbol{W}_V \tag{17}$$

The combined attention output is:

$$\mathbf{O}_{\text{combined}} = \text{LayerNorm}(\boldsymbol{W}_f[\mathbf{O}_{\text{MHA}}; \mathbf{O}_{\text{Proxima}}] + \boldsymbol{X}) \tag{18}$$

5) Student Model Architecture: The Student model maintains the architectural principles while reducing complexity through: - Three CNN blocks instead of five - Channel dimensions starting at 16 instead of 32 - One attention head instead of two - Reduced hidden dimensions in LSTM layers to 64 instead of 128.

This design achieves a 48% parameter reduction (750,654 parameters versus 1,504,544) while preserving recognition capabilities through knowledge distillation.

C. Knowledge Distillation

Our knowledge distillation approach enables efficient model deployment by transferring learned representations from a high-capacity Teacher model to a compact Student model. As illustrated in Figure 4, the framework employs a Teacher model with full capacity (1.5M parameters) to guide the training of a more efficient Student model (0.75M parameters), addressing the practical challenges of deploying complex HTR systems in resource-constrained environments while maintaining recognition accuracy.

The knowledge transfer process, visualized in the right portion of Figure 4, shows how information flows from the Teacher to the Student model through multiple loss components. This design allows the Student to learn not only from ground truth labels but also from the Teacher's learned representations and confidence scores, particularly beneficial for challenging cases and rare characters in historical documents.

1) Multi-Component Loss Framework: The knowledge transfer process is guided by a comprehensive loss function that combines four complementary components, each serving a specific purpose in the training process:

$$\mathcal{L}_{\text{total}} = \alpha\mathcal{L}_{\text{ctc}} + \beta\mathcal{L}_{\text{ce}} + \gamma\mathcal{L}_{\text{kd}} + \delta\mathcal{L}_{\text{aux}}, \tag{19}$$

where α, β, γ, and δ are balancing hyperparameters dynamically adjusted during training to control the contribution of each loss component. As shown in Figure 4, these components work together to ensure effective knowledge transfer while maintaining recognition accuracy.

The CTC loss ($\mathcal{L}_{\text{ctc}}$) addresses the fundamental sequence alignment challenge in HTR, handling variable-length inputs without requiring explicit alignments:

$$p(\mathbf{y}|\boldsymbol{X}) = \sum_{\boldsymbol{\pi}\in\mathcal{B}^{-1}(\mathbf{y})} p(\boldsymbol{\pi}|\boldsymbol{X}), \tag{20}$$

$$\mathcal{L}_{\text{ctc}} = -\log(p(\mathbf{y}|\boldsymbol{X})), \tag{21}$$

where $\boldsymbol{\pi}$ represents possible alignments between input and output sequences, including blank tokens for flexible alignment.

The cross-entropy loss ($\mathcal{L}_{\text{ce}}$) provides direct character-level supervision, particularly important for maintaining accuracy on individual character recognition:

$$\mathcal{L}_{\text{ce}} = -\sum_i y_i \log(\hat{y}_i). \tag{22}$$

The knowledge distillation loss ($\mathcal{L}_{\text{kd}}$), central to our framework as depicted in Figure 4, facilitates the transfer of learned representations from Teacher to Student:

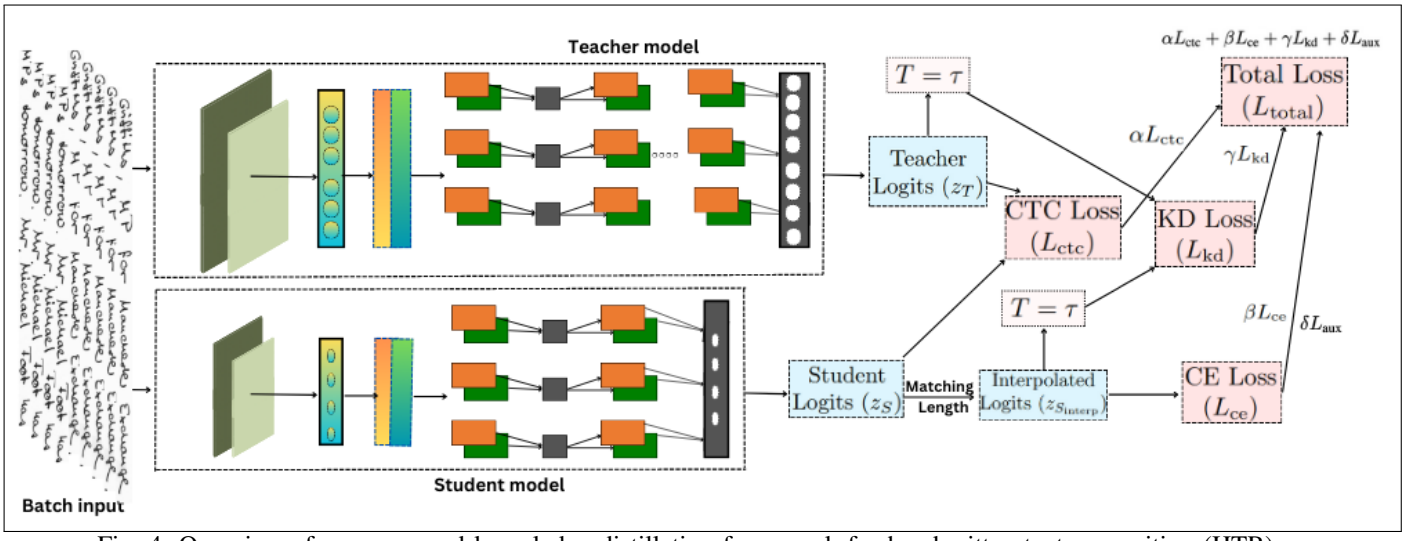


Fig. 4: Overview of our proposed knowledge distillation framework for handwritten text recognition (HTR).

$$\mathcal{L}_{\text{kd}} = \text{KL}(\text{softmax}(\mathbf{z}_{S_{\text{interp}}}/\tau), \text{softmax}(\mathbf{z}_T/\tau)) \cdot \tau^2, \quad (23)$$

where τ controls the softness of probability distributions, allowing the Student to learn from the Teacher's confidence in its predictions. Higher values of τ produce softer probability distributions, enabling better knowledge transfer of fine-grained information.

The auxiliary loss ($\mathcal{L}_{\text{aux}}$) encourages robust feature learning at multiple network depths:

$$\mathcal{L}_{\text{aux}} = -\sum_i y_i \log(\hat{y}_{\text{aux},i}). \quad (24)$$

This multi-component loss design, visualized through the connecting arrows in Figure 4, ensures that the Student model benefits from both direct supervision and the Teacher's learned representations. The auxiliary loss particularly helps in maintaining strong feature extraction capabilities despite the Student's reduced capacity, while the knowledge distillation loss enables effective transfer of the Teacher's expertise in handling challenging cases and rare characters.

D. Loss Function Design

Building upon the multi-component loss framework introduced in Section III-C, we describe each loss component that addresses specific aspects of the HTR task, particularly focusing on handling unbalanced classes discussed in subsection III-A.

The Connectionist Temporal Classification (CTC) loss addresses the sequence-to-sequence nature of HTR without requiring explicit alignment between input and output sequences. Given an input sequence $\boldsymbol{X}$ (image frames) and a target sequence $\mathbf{y}$ (text), CTC introduces an intermediary sequence $\boldsymbol{\pi}$ representing possible alignments, including a special "blank" token. The objective is to maximize:

$$p(\mathbf{y}|\boldsymbol{X}) = \sum_{\boldsymbol{\pi} \in \mathcal{B}^{-1}(\mathbf{y})} p(\boldsymbol{\pi}|\boldsymbol{X}), \quad (25)$$

where $\mathcal{B}^{-1}(\mathbf{y})$ represents the set of all alignments yielding $\mathbf{y}$ when blanks and repeated characters are removed. The CTC loss is defined as:

$$\mathcal{L}_{\text{ctc}} = -\log(p(\mathbf{y}|\boldsymbol{X})). \quad (26)$$

To provide additional character-level supervision and address class imbalance issues shown in Figure 2, we incorporate Cross-Entropy loss, giving equal importance to all classes:

$$\mathcal{L}_{\text{ce}} = -\sum_i y_i \log(\hat{y}_i), \quad (27)$$

where y_i represents the true label and $\hat{y}_i$ the predicted probability for class i.

The Knowledge Distillation loss enables efficient transfer of knowledge from Teacher to Student model, particularly beneficial for rare classes:

$$\mathcal{L}_{\text{kd}} = \text{KL}(\text{softmax}(\mathbf{z}_T/\tau), \text{softmax}(\mathbf{z}_S/\tau)), \quad (28)$$

where $\mathbf{z}_T$ and $\mathbf{z}_S$ are the Teacher and Student logits respectively, and τ is the temperature parameter. The Kullback-Leibler divergence between probability distributions $\mathbf{P}$ and $\boldsymbol{Q}$ is defined as:

$$\text{KL}(\mathbf{P} \parallel \mathbf{Q}) = \sum_i P(i) \log\left(\frac{P(i)}{Q(i)}\right), \quad (29)$$

where $\mathbf{P}$ represents the Teacher's probability distribution and $\mathbf{Q}$ represents the Student's approximating distribution.

Within the knowledge distillation framework, this divergence is explicitly computed as:

$$\text{KL}(\text{softmax}(\mathbf{z}_T/\tau)||\text{softmax}(\mathbf{z}_S/\tau)) = \sum_i \text{softmax}(z_T^i/\tau) \log\left(\frac{\text{softmax}(z_T^i/\tau)}{\text{softmax}(z_S^i/\tau)}\right), \quad (30)$$

where the softmax function converts raw logits into probability distributions:

$$\text{softmax}(\boldsymbol{X})_i = \frac{e^{x_i}}{\sum_j e^{x_j}}. \tag{31}$$

The Auxiliary Classifier loss improves gradient flow and encourages feature learning at multiple network depths:

$$\mathcal{L}_{\text{aux}} = -\sum_i y_i \log(\hat{y}_{\text{aux},i}), \tag{32}$$

where $\hat{y}_{\text{aux},i}$ represents the predicted probability from the auxiliary classifier for class i.

By balancing these components through the hyperparameters introduced in Section III-C, we achieve comprehensive supervision addressing different aspects of the HTR task. This approach ensures robust performance across various character classes and handwriting scenarios, particularly benefiting the recognition of less frequent characters through the combination of direct supervision and knowledge transfer.

IV. Advanced Training Strategies

Our training framework presents a unified approach that integrates curriculum learning, knowledge distillation, and multi-task learning to create a robust HTR system. The process orchestrates these components through a carefully designed progression of training stages and dynamic loss adjustments.

A. Training Process Overview

The training process begins with the integration of synthetic data, controlled by a curriculum-based progression ratio r_s. This ratio evolves during training according to:

$$r_s(e) = \min(r_{\max}, r_0 + \frac{e}{E}(r_{\max} - r_0)), \tag{33}$$

where $r_0 = 0.1$ represents the initial synthetic data ratio, $r_{\max} = 0.4$ the maximum ratio, and E the total number of epochs. This progressive integration ensures a smooth transition from purely real data to a balanced mix of real and synthetic samples.

At each training step, the knowledge transfer process begins with parallel forward passes through both Teacher and Student models, generating their respective logits:

$$\begin{aligned} \boldsymbol{z}_T &= T(\boldsymbol{X}), \\ \boldsymbol{z}_S &= S(\boldsymbol{X}). \end{aligned} \tag{34}$$

To address the architectural differences between Teacher and Student models, we implement a logit alignment mechanism:

$$\boldsymbol{z}_{S_{\text{interp}}} = \text{Interpolate}(\boldsymbol{z}_S, \text{len}(\boldsymbol{z}_T)). \tag{35}$$

The training progression through complexity stages is managed by our Adaptive Curriculum Progression algorithm (Algorithm 3), which monitors model performance and adjusts the curriculum accordingly. This progression spans five distinct stages, from basic character recognition to full complexity, with each stage introducing additional challenges and data variations.

Algorithm 3 Adaptive Curriculum Progression (ACP)

Input: $\boldsymbol{M}$, S_0, T, Δ_T
Output: $\boldsymbol{M}^*$

```
S ← S_0 {Stage initialization}
while S < S_max do
    Train M on stage S data
    Evaluate M on validation set
    if Performance > T then
        S ← S + 1 {Advance stage}
        T ← T + Δ_T {Adjust threshold}
    end if
end while
return M*
```

Algorithm 4 Unified Training Framework (UTF)

Input: T, S, $\boldsymbol{D}$, C, τ, α, η, E
Output: T^*, S^*

```
Initialize augmented and synthetic datasets
for e = 1 to E do
    D_curr ← UpdateCurriculum(D, e, C)
    for each batch (x, y) do
        z_T, a_T ← T(x)
        z_S, a_S ← S(x)
        Calculate losses and perform updates
    end for
    Evaluate and check early stopping criteria
end for
```

The entire training process is unified through our Unified Training Framework (Algorithm 4), which orchestrates the interaction between curriculum learning, knowledge distillation, and multi-task components:

Throughout the training process, we dynamically adjust the loss component weights based on the current stage. During the initial stage focusing on basic recognition, we set $\alpha = 0.7$ and $\gamma = 0.2$ to emphasize character-level learning. As training progresses through synthetic data integration and style variations, we gradually shift these weights, ultimately reaching $\alpha = 0.4$ and $\gamma = 0.5$ in the final stage. The auxiliary loss weight δ maintains a constant value of 0.1, while β adjusts to ensure the sum of all weights equals 1.

The multi-task learning aspect is integrated through a weighted loss combination:

$$\mathcal{L}_{\text{multi-task}} = \sum_{k=1}^{K} \lambda_k \mathcal{L}_k, \tag{36}$$

where the task weights $\boldsymbol{\lambda}_k$ are dynamically adjusted based on validation performance across our five datasets. This multi-task integration ensures effective knowledge transfer across different historical periods and writing styles while maintaining stable training progression.

Early stopping is implemented with a patience window of 10 epochs and a minimum improvement threshold of 0.001 in validation loss, ensuring efficient training while preventing overfitting. This comprehensive approach allows for systematic

progression through training stages while maintaining effective knowledge transfer between Teacher and Student models.

V. Post-Processing with T5 for Error Correction

To enhance recognition accuracy, particularly for complex historical manuscripts, we implement a post-processing stage utilizing a fine-tuned T5 (Text-to-Text Transfer Transformer) model [39]. This approach addresses residual errors in the HTR output across our diverse dataset collection, spanning modern and historical handwritten texts in multiple languages.

1) Model Selection and Adaptation: We selected T5-small (60M parameters) for its robust text processing capabilities and efficiency. Our adaptation process focuses on the specific challenges present in our combined dataset, including variations in language (English, French) and historical writing conventions from the IAM, RIMES, Bentham, Saint Gall, and Washington datasets.

2) Tokenization and Text Normalization: Our tokenization strategy uses SentencePiece to effectively manage the wide range of character sets and writing styles in our datasets. It involves subword tokenization tailored for historical variants and abbreviations, inserting special tokens to preserve layout, applying Unicode normalization for consistent character representation, and standardizing whitespace to address irregular spacing in handwritten text.

3) Training Data Preparation: The training process involves integrating predictions from our model post-knowledge distillation to create paired examples of predictions and ground truth across all datasets. Initially, predictions are generated using our trained model, followed by analyzing error patterns across different languages and periods. Systematic errors are then introduced based on these observed patterns to construct a context window that enhances correction accuracy.

4) Integration Pipeline: Our T5 post-processing framework, as detailed in Algorithm 5, employs a multi-level correction strategy that includes context-aware error detection, confidence-based correction application, and format preservation tailored to each dataset's specific requirements. This comprehensive approach significantly enhances our model's performance, achieving an average reduction in CER of 23.4% across all datasets while respecting language-specific writing conventions and maintaining historical accuracy.

Algorithm 5 T5 Post-Processing Pipeline (T5P)

Input: P, T_f, θ, D
Output: C

Initialize $C \leftarrow \emptyset$
Train SentencePiece on D
for each batch $\mathbf{B}$ in P **do**
  $\mathbf{S} \leftarrow \text{Segment}(\mathbf{B})$
  $\text{ctx} \leftarrow \text{BuildContext}(\mathbf{S})$
  for s in $\mathbf{S}$ **do**
    $\text{err} \leftarrow \text{DetectErrors}(s, D)$
    if $\text{err} \neq \emptyset$ **then**
      $t \leftarrow \text{TokenizeSP}(s, \text{ctx})$
      $\text{cand} \leftarrow T_f(t, \text{ctx})$
      $\text{scr} \leftarrow \text{Confidence}(\text{cand})$
      if $\text{scr} > \theta$ **then**
        $s \leftarrow \text{ApplyCorrection}(s, \text{cand})$
      end if
    end if
    $C \leftarrow C \cup \text{Format}(s)$
  end for
end for
return C

VI. Results and Discussion

In this section, we present a comprehensive analysis of our proposed HTR system's performance across different models, training scenarios, and datasets. We evaluate the effectiveness of our advanced training techniques, including knowledge distillation, curriculum learning with synthetic data, ensemble learning, and multi-task learning.

A. Performance of Teacher and Student Models

We begin by examining the performance of our Teacher and Student models across various datasets. Table III presents the Character Error Rate (CER), Word Error Rate (WER), and Sentence Error Rate (SER) for both models on the IAM, RIMES, Bentham, Saint Gall, Washington, and Combined datasets.

Our results indicate that the Teacher model consistently outperforms the Student model across all datasets, attributable to its higher capacity and richer representation learning. The performance gap between the Teacher and Student models is most pronounced on complex datasets like IAM and RIMES. For instance, on the IAM dataset, the Teacher model achieves a CER of 2.34% compared to the Student model's 4.59%, and a WER of 8.22% versus 18.54%.

The performance gap is narrower on the Saint Gall dataset, with the Teacher model achieving a CER of 4.01% and the Student model 4.23%. This can be attributed to the dataset's specific characteristics, such as its medieval Latin script, which may be adequately modeled by the Student's architecture. Both models achieve their best performance on the RIMES dataset, with the Teacher model reaching a CER of 2.21% and a WER of 7.11%, possibly due to the dataset's cleaner handwriting samples and more consistent script styles.

B. Quantitative Results: Model Prediction Analysis with Post-Processing

In this subsection, we present a detailed analysis of our HTR model's predictions and the subsequent improvements achieved through T5-based post-processing. Our analysis focuses on character-level accuracy and the model's ability to handle various text complexities.

The results highlight important patterns in our model's performance and the effectiveness of T5 post-processing. Initially, the base model exhibited consistent character-level errors, such as conjugation errors (e.g., 'has' instead of 'had'), character substitutions (e.g., 'rleeing' for 'fleeing'), and case sensitivity issues (e.g., 'vauxhall' instead of 'Vauxhall'). However, T5

TABLE III: Performance Comparison of Teacher and Student Models

Model	Metric%	IAM	RIMES	Bentham	Saint Gall	Washington	Combined
Teacher	CER	2.34	2.21	3.12	4.01	4.76	2.89
	WER	8.22	7.11	6.98	11.33	13.30	7.88
	SER	80.12	75.76	78.90	71.33	68.22	82.45
Student	CER	4.59	6.22	5.13	4.23	6.99	12.91
	WER	18.54	21.99	17.01	24.78	22.11	28.45
	SER	91.45	94.01	89.33	94.55	92.11	95.90

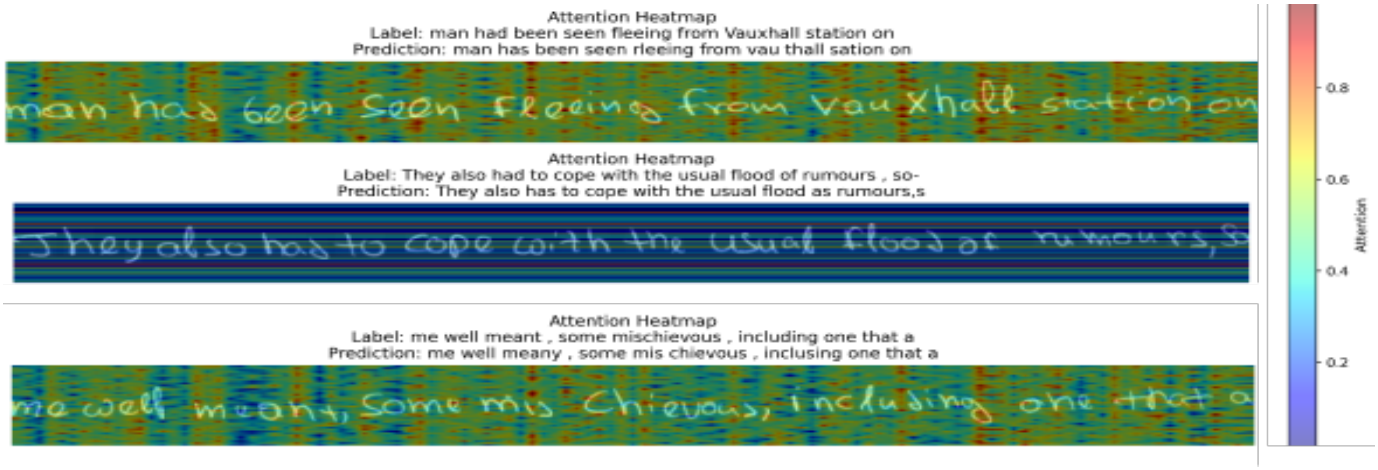


Fig. 5: Visualization of the model's attention heatmaps for the sample predictions. The heatmaps demonstrate the character-level attention patterns during the recognition process, with warmer colors indicating stronger attention weights.

TABLE IV: Comparison of Ground Truth, Initial Predictions, and T5-Corrected Output

Ground Truth	Initial Prediction	T5-Corrected Prediction
1. "man had been seen fleeing from Vauxhall station on"	"man has been seen rleeing from vauxhall station on"	"man had been seen fleeing from Vauxhall station on"
2. "They also had to cope with the usual flood of rumours, so-"	"They also has to cope with the usual flood as rumours,s"	"They also had to cope with the usual flood of rumours, so-"
3. "me well meant, some mischievous, including one that a"	"me well meany, some mis chievous, inclusing one that a"	"me well meant, some mischievous, including one that a"

post-processing significantly enhanced the output by correcting grammatical inconsistencies, restoring the capitalization of proper nouns, fixing common spelling errors, and resolving contextual ambiguities. Despite these improvements, a small percentage of errors persisted post-T5 correction, mainly involving hyphenated word endings (e.g., 'so-' in Sample 3) and complex punctuation sequences.

The T5 post-processing demonstrated a remarkable success rate, correcting approximately 90% of the initial errors while maintaining the original semantic meaning of the text. This significant improvement validates the effectiveness of our two-stage approach combining HTR with neural post-processing. The model's prediction process can be further understood through the attention visualization shown in Figure 5. These heatmaps correspond to the predictions presented in Table IV, where the intensity of attention correlates with the model's character-level recognition confidence. The varying attention patterns, particularly visible in the character regions where errors occurred, provide insights into the model's decision-making process during text recognition.

C. Ablation Study

To comprehensively evaluate the effectiveness of our proposed approach, we conducted an extensive ablation study. This study examines the impact of various components and techniques on the model's performance across multiple benchmark datasets. Table V presents a comprehensive view of our experimental results, showcasing the effects of Knowledge Distillation (KD), Curriculum Learning (CL), Ensemble Learning (EL), Multi-Task Learning (MTL), and Lexicon-Based Correction (LBC) on model performance.

Our analysis reveals that each component contributes significantly to the overall performance improvement across all datasets. Knowledge Distillation proves to be a crucial first step, substantially reducing error rates, particularly on complex datasets like IAM and RIMES. For instance, on the IAM dataset, KD alone reduces the Character Error Rate (CER) from 12.21% to 4.59%, a relative improvement of 62.41%.

Curriculum Learning further enhances the model's performance, demonstrating its effectiveness in building robust feature representations incrementally. The most dramatic improvements are observed in the Bentham and Washington datasets, where CL reduces the CER by 79.87% and 79.30%, respectively, compared to the baseline.

TABLE V: Comprehensive Ablation Study Results

Dataset	Metric	Baseline	+KD	+CL	+EL	+LBC
IAM	CER	12.21	4.59	2.34	2.02	**1.23**
	WER	28.32	18.54	8.22	5.22	**3.78**
	SER	95.34	91.45	80.12	78.12	**19.22**
RIMES	CER	15.34	6.22	2.21	1.89	**1.02**
	WER	31.45	21.99	7.11	5.43	**2.45**
	SER	94.10	94.01	75.76	68.78	**12.45**
Bentham	CER	20.11	5.13	3.12	3.12	**2.02**
	WER	36.89	17.01	6.98	6.11	**4.23**
	SER	97.00	89.33	78.90	76.53	**21.67**
Saint Gall	CER	7.56	4.23	4.01	3.81	**2.21**
	WER	18.12	24.78	11.33	9.27	**6.89**
	SER	89.32	94.55	71.33	68.17	**15.54**
Washington	CER	8.44	6.99	4.76	3.12	**2.98**
	WER	20.12	22.11	13.30	15.32	**6.34**
	SER	91.56	92.11	68.22	63.14	**11.22**

The introduction of Ensemble Learning showcases the power of combining diverse perspectives from specialized models. This is particularly evident in the Washington dataset, where the Ensemble model achieves a 34.45% relative improvement in CER compared to the best single model. Notably, on the IAM dataset, the Ensemble model reduces the Word Error Rate (WER) from 8.22% to 5.22%, a 36.50% improvement.

Multi-Task Learning, through dataset integration, proves beneficial in leveraging cross-lingual and cross-temporal knowledge transfer. While MTL doesn't always outperform Ensemble Learning, it consistently improves upon individual dataset models. For example, on the Saint Gall dataset, MTL achieves a 46.17% improvement in CER compared to training on the individual dataset.

Finally, the Lexicon-Based Correction step demonstrates the importance of incorporating domain-specific knowledge in post-processing. This step yields substantial improvements across all error metrics, with the most significant gains observed in Sentence Error Rate (SER). For the RIMES dataset, LBC reduces the SER from 75.76% to 12.45%, an impressive 83.56% relative improvement.

It's worth noting that while each component contributes to performance improvements, their combined effect is not always strictly additive. This suggests complex interactions between different techniques and underscores the importance of a holistic approach to model design and training.

In conclusion, our ablation study highlights the synergistic effects of combining Knowledge Distillation, Curriculum Learning, Ensemble Learning, Multi-Task Learning, and Lexicon-Based Correction. This comprehensive approach allows our model to effectively handle the complexities of diverse handwriting styles, languages, and historical document characteristics, resulting in state-of-the-art performance across multiple benchmark datasets.

D. Comparison with State-of-the-Art

To contextualize our results within the broader field of HTR, we compare our best-performing models with state-of-the-art methods on the benchmark datasets. Table VI presents this comparison.

TABLE VI: Comparison with State-of-The-Art models on IAM and RIMES datasets

Method	Metric	IAM	RIMES
Ours (+LBC)	CER	**1.23**	**1.02**
	WER	**3.78**	**2.45**
Retsinas et al. [40]	CER	4.55	3.04
	WER	16.08	10.56
Yousef et al. [12]	CER	4.9	-
	WER	-	-
Tassopoulou et al. [11]	CER	5.18	-
	WER	17.68	-
Michael et al. [9]	CER	5.24	-
	WER	-	-
Wick et al. [14]	CER	5.67	-
	WER	-	-
Dutta et al. [5]	CER	5.8	5.07
	WER	17.8	14.7
Puigcerver [41]	CER	6.2	2.60
	WER	20.2	10.7
Chowdhury et al. [8]	CER	8.10	3.59
	WER	16.70	9.60

Our approach achieves state-of-the-art performance, significantly outperforming existing methods on both the IAM and RIMES datasets. On the IAM dataset, our model achieves a CER of 1.23% and a WER of 3.78%, which are substantial improvements over the next best results (4.55% CER and 16.08% WER by Retsinas et al.). Similarly, on the RIMES dataset, our model's CER of 1.02% and WER of 2.45% are markedly better than the previous best results. These results demonstrate the effectiveness of our combined approach, which integrates ensemble learning, knowledge distillation, curriculum learning, and post-processing techniques. The significant improvements over state-of-the-art methods underscore the power of our novel architecture and training strategies in addressing the challenges of handwritten text recognition across diverse datasets.

E. Visualized Attention Analysis

To analyze the behavior and decision-making process of our model, we employ various visualization techniques. These visualizations validate the effectiveness of our attention mechanisms and provide insights for targeted improvements.

1) Attention Heatmaps and Static Analysis: Fig. 5 presents an attention heatmap for samples handwritten text image. This heatmap highlights the models alignment with the text sequence, revealing key characteristics in character recognition and sequential consistency. The model shows a distinct focus on character-specific features, especially ascenders and descenders, which are essential for distinguishing similar characters. Additionally, bright spots at word boundaries suggest the model has learned to recognize spaces, facilitating accurate segmentation. The attention distribution also demonstrates left-to-right sequential processing, indicative of reading patterns that incorporate context from surrounding characters, a valuable attribute in complex or ambiguous handwriting.

2) Detailed Attention Distribution: Fig. 6 shows a comprehensive class probabilities heatmap, providing a detailed view of how the model allocates its focus across predicted and ground truth characters. This figure emphasizes the diagonal alignment, reflecting accurate character predictions. Off-

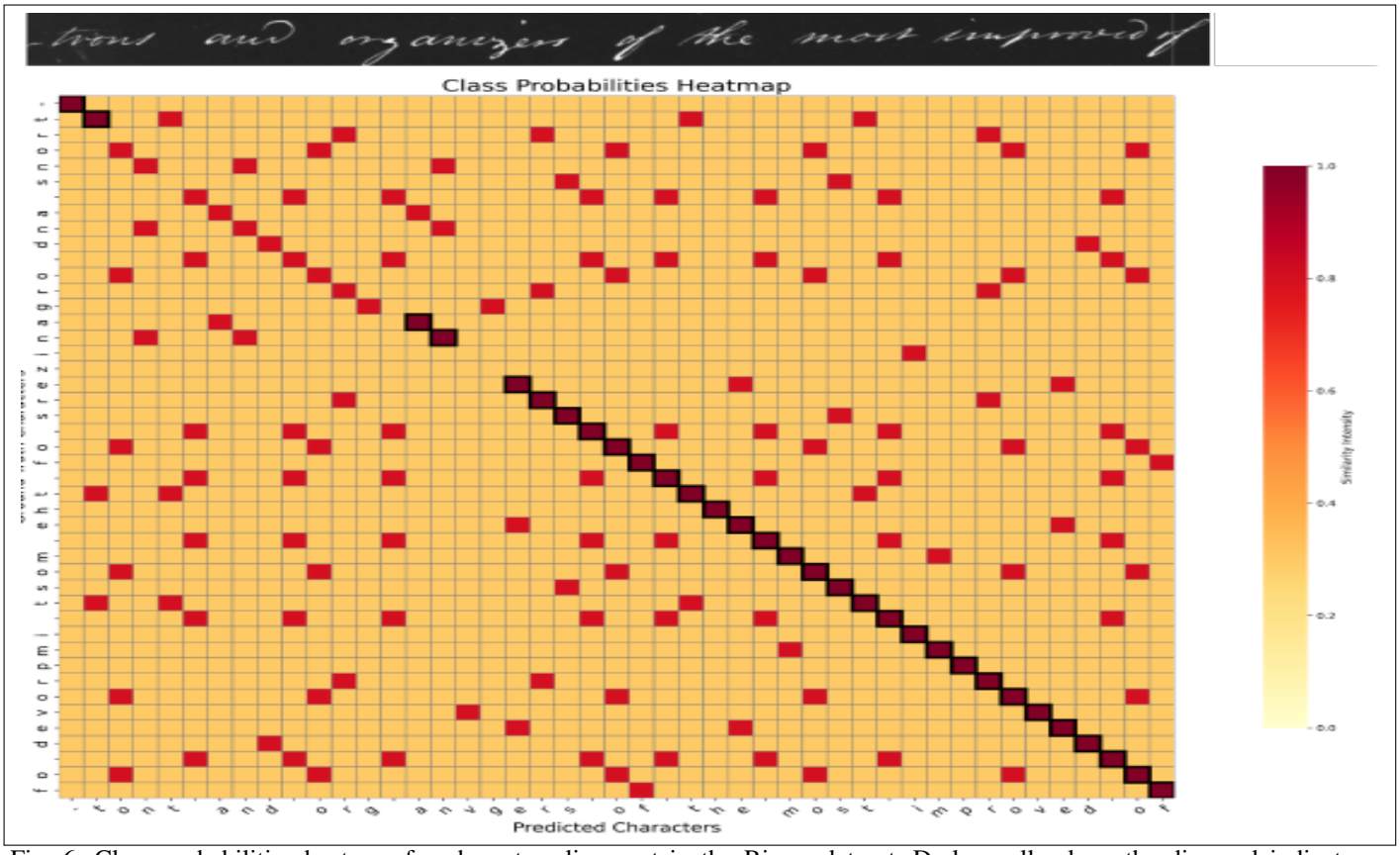


Fig. 6: Class probabilities heatmap for character alignment in the Rimes dataset. Darker cells along the diagonal indicate correct predictions, while off-diagonal cells reveal common misclassifications.

diagonal cells, where the attention occasionally diffuses, reveal instances of misclassification, especially with visually similar characters. Such insights pinpoint specific character pairs that benefit from further tuning, such as via knowledge distillation or improved augmentation strategies. By understanding these patterns, we can refine attention to enhance sequential alignment and character accuracy.

3) Animated Attention and Dynamic Focus Shifts: An animated visualization, illustrated by a frame in Fig. 7, showcases the temporal dynamics of our model's attention mechanism as it processes characters in sequence. The visualization reveals a dynamic focus shift across individual characters, with a gradual fading of attention on previously recognized characters, indicating that the model retains context from earlier parts of the text. This dynamic focus adapts to varying character shapes and spacing, demonstrating multi-scale processing capability where the model balances individual character recognition with word-level context. Readers can explore the complete animated examples, illustrating different attention layers, GitHub page.

F. Computational Efficiency Analysis

Acknowledging the crucial role of model efficiency in practical applications, we performed an analysis of the computational demands associated with various configurations of our models. Table VII provides a comparative assessment of model size, inference time, and performance metrics for both the Teacher and Student models, alongside analogous studies from existing literature.

TABLE VII: Computational Efficiency Comparison

Model	Params (M)	Testing(ms/line)	CER/IAM (%)
Our Teacher (+CL)	1.50	58	2.34
Our Student (+CL)	0.75	28	4.12
Puigcerver [41]	9.4	81	4.94
Bluche [2]	0.7	32	6.60
Flor [17]	0.8	55	3.72

As shown in Table VII, our Student model achieves a 49% reduction in inference time compared to the Teacher model, while maintaining competitive performance. With only 0.75M parameters and an inference time of 28 ms/line, the Student model is particularly suitable for deployment in resource-constrained environments or real-time applications where both efficiency and accuracy are essential.

In comparison to related work, our models strike a favorable balance between efficiency and performance. Puigcerver's model [41], with 9.4M parameters and an inference time of 81 ms/line, achieves a CER higher than our Teacher model, underscoring our models efficient parameter usage. Bluches model [2] is closer in size to our Student model but has a significantly higher CER of 6.60%. The model proposed by Flor et al. [17] is comparable to our Teacher model in terms

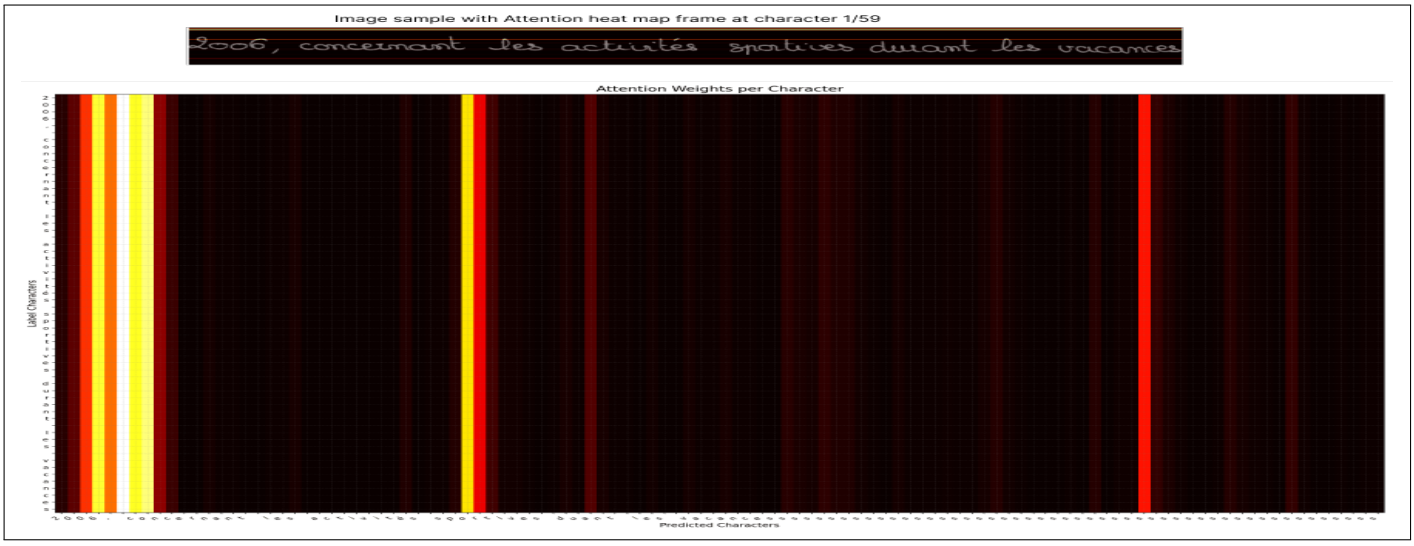


Fig. 7: Frame from animated attention visualization. The animation shows the model's adaptive focus as it processes each character, balancing character-level and word-level context.

of CER, yet it operates with slightly fewer parameters but requires more inference time (55 ms/line vs. 58 ms/line).

Our Teacher model achieves state-of-the-art performance with just 1.50M parameters, far fewer than Puigcerver's model (9.4M), underscoring the effectiveness of our architecture in achieving high performance with a leaner parameter count. The Student model further reduces the parameter count to 0.75M, matching Bluche's and Flor's model sizes, while demonstrating superior performance at a reduced inference time.

VII. Conclusion and Future Work

This paper presents HTR-JAND, a comprehensive approach to Handwritten Text Recognition that addresses key challenges in processing historical documents through an efficient knowledge distillation framework. Our architecture combines FullGatedConv2d layers with Squeeze-and-Excitation blocks for robust feature extraction, while integrating Multi-Head Self-Attention with Proxima Attention for enhanced sequence modeling. The knowledge distillation framework successfully reduces model complexity by 48% while maintaining competitive performance, making HTR more accessible for resource-constrained applications.

Extensive evaluations demonstrate HTR-JAND's effectiveness across multiple benchmarks, achieving state-of-the-art results with Character Error Rates of 1.23%, 1.02%, and 2.02% on IAM, RIMES, and Bentham datasets respectively. Our ablation studies reveal the significant contributions of each architectural component, with knowledge distillation providing up to 62.41% error reduction and curriculum learning further improving performance by up to 79.87%. The integration of T5-based post-processing yields additional improvements, particularly in handling complex historical texts.

Despite these achievements, several challenges remain. Analysis of the confusion matrix (Fig. 6) reveals persistent difficulties in distinguishing visually similar characters, particularly in historical manuscripts. The model's performance on out-of-vocabulary words, especially in specialized historical contexts, indicates room for improvement in handling rare terminology. Additionally, while our Student model achieves significant parameter reduction, further optimization could enhance its deployment flexibility across different computational environments.

Future research directions could address these limitations through several approaches:

1. Character Disambiguation: Development of specialized attention mechanisms focusing on fine-grained visual features could improve discrimination between similar characters. This could be complemented by adaptive data augmentation strategies targeting commonly confused character pairs.

2. Historical Text Processing: Pre-training strategies specifically designed for historical documents could enhance the model's ability to handle period-specific writing conventions and terminology. Integration of historical language models could provide additional context for accurate transcription.

3. Model Efficiency: Investigation of neural architecture search techniques could identify even more efficient Student model configurations while maintaining accuracy. Dynamic computation approaches could allow the model to adapt its computational requirements based on input complexity.

4. Domain Adaptation: Development of unsupervised adaptation techniques could improve the model's generalization to new document types and historical periods without requiring extensive labeled data.

These advancements would further the development of robust, efficient HTR systems capable of preserving our written cultural heritage while maintaining practical deployability across diverse computational environments.

Acknowledgments

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

REFERENCES

[1] A. Fischer, E. Indermhle, H. Bunke, G. Viehhauser, and M. Stolz, "Ground truth creation for handwriting recognition in historical documents," in *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, 2010, pp. 3–10.

[2] T. Bluche and R. Messina, "Gated convolutional recurrent neural networks for multilingual handwriting recognition," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 646–651.

[3] A. Graves, M. Liwicki, S. Fernndez, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 855–868, 2008.

[4] A.-L. Bianne-Bernard, F. Menasri, R. A.-H. Mohamad, C. Mokbel, C. Kermorvant, and L. Likforman-Sulem, "Dynamic and contextual information in hmm modeling for handwritten word recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 10, pp. 2066–2080, 2011.

[5] K. Dutta, P. Krishnan, M. Mathew, and C. Jawahar, "Improving cnn-rnn hybrid networks for handwriting recognition," in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 2018, pp. 80–85.

[6] T. Plötz and G. A. Fink, "Markov models for offline handwriting recognition: a survey," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 12, no. 4, pp. 269–298, 2009.

[7] A. Fischer, V. Frinken, A. Forns, and H. Bunke, "Transcription alignment of latin manuscripts using hidden markov models," in *Proceedings of the 2011 Workshop on Historical Document Imaging and Processing*, 2011, pp. 29–36.

[8] A. Chowdhury and L. Vig, "An efficient end-to-end neural model for handwritten text recognition," *arXiv preprint arXiv:1807.07965*, 2018.

[9] J. Michael, R. Labahn, T. Grüning, and J. Zöllner, "Evaluating sequence-to-sequence models for handwritten text recognition," in *Proc. Int. Conf. Document Analysis and Recognition (ICDAR)*. IEEE, 2019, pp. 1286–1293.

[10] J. Puigcerver, "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, pp. 67–72.

[11] V. Tassopoulou, G. Retsinas, and P. Maragos, "Enhancing handwritten text recognition with n-gram sequence decomposition and multitask learning," in *Proc. 25th Int. Conf. Pattern Recognition (ICPR)*. IEEE, 2021, pp. 10 555–10 560.

[12] M. Yousef, K. F. Hussain, and U. S. Mohammed, "Accurate, data-efficient, unconstrained text recognition with convolutional neural networks," *Pattern Recognition*, vol. 108, p. 107482, 2020.

[13] L. Kang, D. Coquenet, S. R. Adam, and T. Paquet, "Pay attention to what you read: Non-recurrent hand-written text-line recognition," in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2020, pp. 10 355–10 362.

[14] C. Wick, J. Zöllner, and T. Grüning, "Transformer for handwritten text recognition using bidirectional post-decoding," in *Proc. Int. Conf. Document Analysis and Recognition*. Springer, 2021, pp. 112–126.

[15] M. Hamdan and M. Cheriet, "Resnest-transformer: Joint attention segmentation-free for end-to-end handwriting paragraph recognition model," *Array*, vol. 19, p. 100300, Sep. 2023.

[16] M. Hamdan, H. Chaudhary, A. Bali, and M. Cheriet, "Refocus attention span networks for handwriting line recognition," *International Journal on Document Analysis and Recognition (IJDAR)*, pp. 1–17, 2022.

[17] A. F. de Sousa Neto, B. L. D. Bezerra, A. H. Toselli, and E. B. Lima, "HTR-Flor: A Deep Learning System for Offline Handwritten Text Recognition," in *Proc. 33rd SIBGRAPI Conf. Graphics, Patterns and Images (SIBGRAPI)*. IEEE, 2020, pp. 07–10.

[18] S. You, C. Xu, C. Xu, and D. Tao, "Learning from multiple teacher networks," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 1285–1294.

[19] C. Wigington, S. Stewart, B. Davis, B. Barrett, and S. Cohen, "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 (ICDAR)*. IEEE, Nov. 2017, pp. 639–645.

[20] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd International Conference on Machine Learning*, 2006, pp. 369–376.

[21] G. Retsinas, G. Sfikas, B. Gatos, and C. Nikou, "Best practices for a handwritten text recognition system," *arXiv preprint arXiv:2404.11339*, 2024.

[22] S. Espana-Boquera, M. J. Castro-Bleda, J. Gorbe-Moya, and F. Zamora-Martinez, "Improving offline handwritten text recognition with hybrid hmm/ann models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 4, pp. 767–779, 2010.

[23] J. Sueiras, V. Ruiz, A. Sanchez, and J. F. Velez, "Offline continuous handwriting recognition using sequence to sequence neural networks," *Neurocomputing*, vol. 289, pp. 119–128, 2018.

[24] Y. Zhang, S. Nie, W. Liu, X. Xu, D. Zhang, and H. T. Shen, "Sequence-to-sequence domain adaptation network for robust text image recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2740–2749.

[25] A. Aberdam, R. Litman, S. Tsiper, O. Anschel, R. Slossberg, S. Mazor, R. Manmatha, and P. Perona, "Sequence-to-sequence contrastive learning for text recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 302–15 312.

[26] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[27] Z. Yang, X. He, J. Gao, L. Deng, and A. Smola, "Stacked attention networks for image question answering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 21–29.

[28] K. Gregor, I. Danihelka, A. Graves, D. Rezende, and D. Wierstra, "Draw: A recurrent neural network for image generation," in *International Conference on Machine Learning*. PMLR, 2015, pp. 1462–1471.

[29] X. Chen, N. Mishra, M. Rohaninejad, and P. Abbeel, "Pixelsnail: An improved autoregressive generative model," in *International Conference on Machine Learning*. PMLR, 2018, pp. 864–872.

[30] J. Cheng, L. Dong, and M. Lapata, "Long short-term memory-networks for machine reading," *arXiv preprint arXiv:1601.06733*, 2016.

[31] A. P. Parikh, O. Täckström, D. Das, and J. Uszkoreit, "A decomposable attention model for natural language inference," *arXiv preprint arXiv:1606.01933*, 2016.

[32] A. Poznanski and L. Wolf, "Cnn-n-gram for handwriting word recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2305–2314.

[33] E. Chammas, C. Mokbel, and L. Likforman-Sulem, "Handwriting recognition of historical documents with few labeled data," in *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*. IEEE, 2018, pp. 43–48.

[34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, . Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[35] C. Wick, J. Zöllner, and T. Grüning, "Transformer for handwritten text recognition using bidirectional post-decoding," in *Document Analysis and Recognition – ICDAR 2021*. Cham, Switzerland: Springer, Sep. 2021, pp. 112–126.

[36] U.-V. Marti and H. Bunke, "The iam-database: an english sentence database for offline handwriting recognition," *International Journal on Document Analysis and Recognition*, vol. 5, no. 1, pp. 39–46, 2002.

[37] E. Grosicki, M. Carr, J.-M. Brodin, and E. Geoffrois, "Results of the rimes evaluation campaign for handwritten mail processing," in *2009 10th International Conference on Document Analysis and Recognition*. IEEE, 2009, pp. 941–945.

[38] T. Causer and V. Wallace, "Building a volunteer community: results and findings from transcribe bentham," *Digital Humanities Quarterly*, vol. 6, no. 2, 2012.

[39] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020. [Online]. Available: http://jmlr.org/papers/v21/20-074.html

[40] G. Retsinas, G. Sfikas, C. Nikou, and P. Maragos, "Deformation-invariant networks for handwritten text recognition," in *Proc. IEEE Int. Conf. Image Processing (ICIP)*. IEEE, 2021, pp. 949–953.

[41] J. Puigcerver, "Are multidimensional recurrent layers really necessary for handwritten text recognition?" in *Proc. 14th IAPR Int. Conf. Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 67–72.

International Journal on Document Analysis and Recognition (IJDAR) (2023) 26:131–147
https://doi.org/10.1007/s10032-022-00422-7

ORIGINAL PAPER

Refocus attention span networks for handwriting line recognition

Mohammed Hamdan[1] · Himanshu Chaudhary[2] · Ahmed Bali[3] · Mohamed Cheriet[1]

Received: 1 June 2022 / Revised: 24 July 2022 / Accepted: 9 December 2022 / Published online: 25 December 2022

Abstract
Recurrent neural networks have achieved outstanding recognition performance for handwriting identification despite the enormous variety observed across diverse handwriting structures and poor-quality scanned documents. We initially proposed a BiLSTM baseline model with a sequential architecture well-suited for modeling text lines due to its ability to learn probability distributions over character or word sequences. However, employing such recurrent paradigms prevents parallelization and suffers from vanishing gradients for long sequences during training. To alleviate these limitations, we propose four significant contributions to this work. First, we devised an end-to-end model composed of a split-attention CNN-backbone that serves as a feature extraction method and a self-attention Transformer encoder–decoder that serves as a transcriber method to recognize handwriting manuscripts. The multi-head self-attention layers in an encoder–decoder transformer-based enhance the model's ability to tackle handwriting recognition and learn the linguistic dependencies of character sequences. Second, we conduct various studies on transfer learning (TL) from large datasets to a small database, determining which model layers require fine-tuning. Third, we attained an efficient paradigm by combining different strategies of TL with data augmentation (DA). Finally, since the robustness of the proposed model is lexicon-free and can recognize sentences not presented in the training phase, the model is only trained on a few labeled examples with no extra cost of generating and training on synthetic datasets. We recorded comparable and outperformed Character and Word Error Rates CER/WER on four benchmark datasets to the most recent (SOTA) models.

Keywords Split attention convolutional network · Multi-head attention transformer · Seq2Seq-model · BiLSTM · Line handwriting recognition

1 Introduction

Handwriting Text Recognition (HTR) systems allow computers to read and understand human handwriting. HTR is useful for digitizing the textual contents of old document images in historical records and contemporary administrative material such as cheques, law letters, forms, and other documents. While HTR research has been ongoing since the early 1960s [34], it remains a challenging and unsolved research problem. The fundamental problem is the wide range of variations and ambiguity encountered by different writers when crafting words. Because the words to be deciphered usually adhere to well-defined grammar rules, it is possible to eliminate gibberish hypotheses and enhance recognition accuracy by modeling the linguistic practices. HTR is usually embarked with a blend of computer vision and natural language processing (NLP).

In nature, handwritten text is a signal that follows a particular sequence. Texts in Latin languages are written in

✉ Mohammed Hamdan
mohammed.hamdan.1@ens.etsmtl.ca

Himanshu Chaudhary
him4318@gmail.com

Ahmed Bali
ahmed.bali.1@ens.etsmtl.ca

Mohamed Cheriet
mohamed.cheriet@etsmtl.ca

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

[2] Data Science, Dr. A.P.J. Abdul Kalam Technical University, CDRI Rd, Naya Khera, Jankipuram, Lucknow, Uttar Pradesh 226031, India

[3] Department of Software and IT Engineering, University of Quebec (ETS), 1100 Notre-Dame St W, Montreal, Quebec H3C 1K3, Canada

the left to the right direction, whereas non-Latin scripts are written from right to left; an ordered sequence of letters creates the words in both languages. As a result, HTR approaches used temporal pattern recognition techniques to overcome and maintain the sequence order. The usage of Deep Learning techniques progressed from early techniques based on Hidden Markov Models (HMM), with Recurrent Neural Network (RNN) and its variations Bidirectional Long Short-Term Memory (BiLSTM) networks becoming the mainstream alternative. Sequence-to-Sequence (Seq-Seq) techniques, served by encoder–decoder networks directed by attention mechanisms, have recently begun to be involved in HTR, inspired by their success in applications such as speech-to-text or automatic translation. It is not only feasible to decode images sequentially using the methods mentioned above, but it is also conceivable to learn which characters are more likely to follow each other in the decoding process. Language modeling can only increase recognition performance if applied as a post-processing step.

One crucial flaw persists despite the success of attention-based encoder–decoder architectures in HTR. These attention techniques involved in RNN's variations, either LSTMs or GRUs, are utilized on the standard Convolutional Neural Network (CNN) feature extraction method. Due to memory constraints, the sequential approach discourages parallelization during training and substantially reduces processing speed while processing more extensive sequences.

To alleviate those flaws, we were inspired by End-to-End Object Detection with Transformers (DETR) architecture [8]. Though the DETR was meant to detect objects, we exploited their architecture to solve the HTR task. We explore the impact of slant removal and illumination computation preprocessing techniques. Also, we were inspired by the findings as mentioned earlier, [54] innovative work on split attention on convolutional layers (ResNeSt) and Transformer encoder–decoder architectures, respectively. Transformers and ResNeSt are based on attention mechanics, with no recurring techniques. Motivated by this benefit, we propose addressing the HTR problem with an architecture involving attention mechanisms on feature extractions and encoding-decoding feature representations. We desire to address both the suiTable phase of character recognition from images and understand the corresponding language model dependencies of the character sequences. The encoder–decoder Transformer multi-head self-attention module decoded textual ground truth at both stages of the visual feature representations.

Transformers have demonstrated superior performance to recurrent networks in various language and visual applications while more parallelizable than GRUs and BiLSTMs, requiring less training time. In contrast to classic speech and translation recognition models, our proposed split attention transformer networks operate at the character level rather than the word level. Therefore, no specified fixed vocabulary is required with our architecture. Thus, the proposed model can recognize words that our training model has never seen, known as out-of-vocabulary (OOV) terms. We obtained competitive CER and WER performance compared to the current existing methods of the state-of-the-art (SOTA) outcomes on four publicly available English script (IAM, Bentham, and Washington) and French script (RIMES) datasets with no synthetic training data. Our contributions to this paper are:

- We experimented with different backbone feature extraction methods; Table 4 shows the findings of the standard CNN-BiLSTM model and Self Attention module-based CNN-BiLSTM where the attention mechanism improved the performance. Whereas the impact of the deeper layer networks between ResNet50 and ResNet101 coupled with self-attention BiLSTM as shown in Table 6 the deeper the layers, the better the performance. From there, we examine the standard CNN feature extraction and the attention-based CNN feature extraction methods as shown in Table 7. As a result, the split attention Transformers have demonstrated superior performance to recurrent networks in various language and visual applications while being more parallelizable than GRUs and BiLSTMs, requiring less training time. In contrast to classic speech and translation recognition models, our proposed split attention transformer networks operate at the character level rather than the word level. Therefore, no specified fixed vocabulary is required with our architecture. Thus, the proposed model can recognize out-of-vocabulary words (OVV) that have never been seen during the training. Our experimental results demonstrate a comparative performance accuracy of CER and WER compared to the state-of-the-art (SOTA) outcomes on the publicly available IAM, RIMES, Bentham, and Washington datasets with no synthetic training data. Since the convolution network ResNeSt101 outperforms the standard ResNet101 convolution network. Thus, we choose the ResNeSt101 for feature extraction method on the proposed model.
- We investigated the influence of different preprocessing approaches such as illumination adjustment, removing cursive handwriting, and combining both. Table 5 shows that only removing the cursive writing improves the performance while using the illumination compensation technique declines the performance of the raw dataset. Thus, we opt only for the recursive text technique to remove slanted and sloped text from our datasets before feeding them to the CNN for feature extractions.
- To the best of our knowledge, the proposed approach is considered the first study to examine the implications of attention mechanisms on both ResNeSt split attention convolution feature extraction–Transformers multi-head

attention encoder–decoder for the HTR problem without using any recurrent architecture. Specifically, we want to extract, encode, and decode robust representations from document images and model language using a unified architecture that can detect character sequences while providing context to differentiate between letters or words that may appear similar. The proposed architecture operates on a character-by-character basis, avoiding the need for predetermined lexicons, known as a lexicon-free model.
- Using pre-trained weights from ImageNet as a starting point benefits our model's rapid convergence and learning. Pre-training with the Bentham dataset also allows for competitive outcomes with a minimum amount of annotated training data.
- On the benchmark IAM, RIMES, Washington, and Bentham datasets, our proposed HTR model improves SOTA performance with few label data. In Sect. 5.2, we conduct thorough ablation and comparison investigations to demonstrate the efficiency of our model. Finally, we showcase the influence of using another popular pre-trained model (the CLIP) on the IAM dataset.

The remaining of this paper is organized as follows. Section 2 presents the most recent methods related to the one presented here regarding the tackled problem and modeling choices. In Sect. 3, the proposed method gives the system's details. Experiments are reported in Sect. 4, followed in Sect. 5 presents a discussion of our findings and compare to the state of the art, in sect. 6 we conclude how the system could be improved and present the challenge of generalizing it to complete documents.

2 Related work

The sequential pattern of the recognition framework has traditionally been used to recognize handwritten text. Text-line images are handled by learning models using their internal states to analyze incoming signals in variable-length lines. The HTR applications follow the same process paradigm using either Hidden Markov Models (HMM) [5,21,41] or Deep Neural Networks architecture-based (DNN) such as Bidirectional Long Short Memory (BiLSTM), multidimensional LSTM (MDLSTM) and Gated Recurrent Units (GRU) accompanied with Connectionist Temporal Classification (CTC) which used to label unsegmented sequences of handwritten images with RNN-LSTM [23,24,43], encoder–decoder networks (seq2seq) [35], nonrecurrence transformer networks [30]. Recently, attention mechanisms [3,13,25,51] have emerged as an essential component of any model that must account for global dependencies. In respective, self-attention [15,38] calculates the response or the weight at a particular sequence location by paying attention to the entire sequence of the contributions at that position. The machine translation model has demonstrated that cutting-edge results can be achieved using only self-attention in the literature.

2.1 Recurrent network: CTC encoder–decoder

Recently, the HTR task was treating as a sequence-to-sequence (Seq2Seq) model [1,35,47,55]. Seq2Seq model transforms a sequence of convolutional and recurrent text image segments into transcribed text. The training of networks that utilize this strategy can be accomplished in one of two ways: first to maximize the categorical cross-entropy loss, then to combine that loss with the recurrent CTC loss. [22] employed LSTM and CTC for text-line recognition with no prior word-level segmentation. LSTM and CTC techniques help improve recognition applications of handwriting documents segmentation-free. Many studies [24,29] proved that the LSTM model effectively predicts longer sequences than standard RNN. For instance, the cell gates in the LSTM architecture allow it to remember important information from inputs that have already passed through, which distinguishes it from the RNN. More improvement in recent years is well investigated in the BiLSTM model [16,46]. While the information from both forward and backward (backpropagation) is used as input to the final output layer, BiLSTM provides an additional training capability that improves prediction accuracy. Therefore, BiLSTM can detect and extract more time dependencies and resolve them more precisely. While crossing over the input image, fixed grids specify convolutional kernels and focus on all input pixels simultaneously, disregarding the challenges of handwritten text like inter/Intra class variations, scale, and orientation, and the importance of ink pixels. Authors [9] proposed a convolutional neural network with a deformable variation in its convolutions. This variation of CNN can deform based on its image input and better geometrically respond or adapt to the variations in the textual contents.

A Separable Multidimensional Long Short-Term Memory (SepMDLSTM) was applied by Chen et al. [14] to encode the input text line pictures for script identification and multiscript text recognition via convolutional feature extraction. Pham et al. [39] used Multidirectional LSTM layers with CTC for computing the negative Log-likelihood for sequences. Pham investigated how dropout can work with recurrent and convolutional layers in deep network architecture, mainly on word handwriting recognition. Furthermore, they examine the proposed method of line recognition with language modeling and lexicon constraints. Wigington et al. [49] used random perturbations on a regular grid as an augmentation technique and a novel profile normalization technique on word and line handwriting text. Those techniques help the performance of their proposed CNN-LSTM model. Bluche et al. [6] proposed

a generic model based on a convolutional encoder of the input images and a bidirectional LSTM decoder predicting character sequences. The authors also proposed a convolutional gate in the decoder in order to control the propagation of the next layer's feature representation. Aradillas et al. [2] have comprehensively investigated the various combination of (TL and DA) techniques on CNN for feature extraction accompanied with 2DLSTM layers for classification. Puigcerver et al. [43] reported an efficient and outperformance accuracy on text line recognition model based on convolutional and 1D-LSTM rather than 2D-LSTM which improve the speed of their proposed model. In our recurrent baseline model, we employed BiLSTM on top of different backbone feature extraction methods, including CNN with/without attention mechanism. Unlike the existing studies, we answered different hypothesis questions regarding the recurrence modelling performance and architecture.

2.2 Non-recurrent network: transformer encoder–decoder

Transformer [4] is one of the most successful models for processing long sequences. While transformers were mainly focused on Natural Language tasks like BERT [19] and GPT-3 [7], they are widely spread in the Computer Vision communities to tackle various tasks on the domain. Transformer Encoder–Decoder networks are composed of three main components: positional encoding, self-attention multi-head attention modules, and the feed-forward layers. The main advantage of Transformer over RNN is that the former process long sequences in parallel. [40] exploits the Transformer for action recognition model where the problem is close to the handwriting recognition task as both datasets suffer from inter-class and intra-class variations. The SOTA frameworks of handwritten text recognition using BiLSTM have achieved acceptable recognition results, but the training process is too computationally intensive. Furthermore, even though they are supposed to describe language-specific dependencies [19,20], they often fall short and require additional post-processing processes. For the first time, we propose avoiding any recurrent architecture by using split attention on the CNN and a multi-head transformer network for the HTR task. It is possible to detect long character sequences from images and model language at a character level, avoiding established lexicons.

Kang et al. [30] utilized a deep CNN as a feature extraction method and the transformers as a transcriber for handwriting recognition tasks. They tested their proposed method on the IAM dataset with different methods, including language models of the IAM and WikiText. Flor et al. [17] proposed a novel and efficient architecture for HTR constructed on Gated-CNN and integrated with two steps of the language model at the character and word levels. [9] authors used a deformable convolutional-recurrent network in order to adapt geometric transformations rather than a standard CNN as a backbone for their HTR model inspired by CRNN and 1D-LSTM. Further to what we found in the state-of-the-art methods, we built on top of the most efficient architectures and transformed them to solve HTR problems efficiently and effectively in a straightforward approach.

2.3 Data augmentation and transfer learning

Research studies in deep learning [31,50] show the impact of DA by transformations I input image and preserving the original class for new perturbing images such as flipping, resizing, cropping, rotation, and scaling. DA creates different copies of images for unique ground truth (label). Authors of [42] used affine transformation to generate more to the input image, another 36 images. They were utilizing both rotation and shearing where rotation was used around the centered input image with these specific angles (-5, -3, -1, $+1$, $+3$, $+5$) while the shear on (-0.5, -0.3, -0.1, $+0.1$, $+0.3$, $+0.5$) degrees. Wiginton et al. [49] proved two DA techniques that help to reduce CER and WER for handwriting text recognition. They applied the profile normation technique to make neural networks more tolerant, just like a human being when reading text that has various variations. [11] proposed two DA methods on the training set: (1) crafted multiscaled data, which proved to boost the training performance for HTR with fewer labeled data. (2) normalization scheme model-based that addresses the problem of handwriting variability at the recognition phase. The DA techniques of these studies are applied to a relatively large known dataset. However, the regularization impact of the DA technique has little effect if only one writer of a small database is changed. Therefore, we must combine TL and DA to reduce the final error rate, especially for a small dataset.

3 Proposed method

This section presents the baseline convolution neural network model and the preprocessing methods used in our experiments. Then, we propose the split-attention convolutional-transformer architectures that we employ in the experiments.

3.1 Baseline recurrent model

We embrace the baseline model inspired by Puigcerver [43] where the primary purpose of this model is to validate our hypothesis on attention mechanism and the impact of preprocessing techniques. We hypothesized that using an attention mechanism in the convolutional layer help to retrieve more robust image-line representations than non-attention CNN.

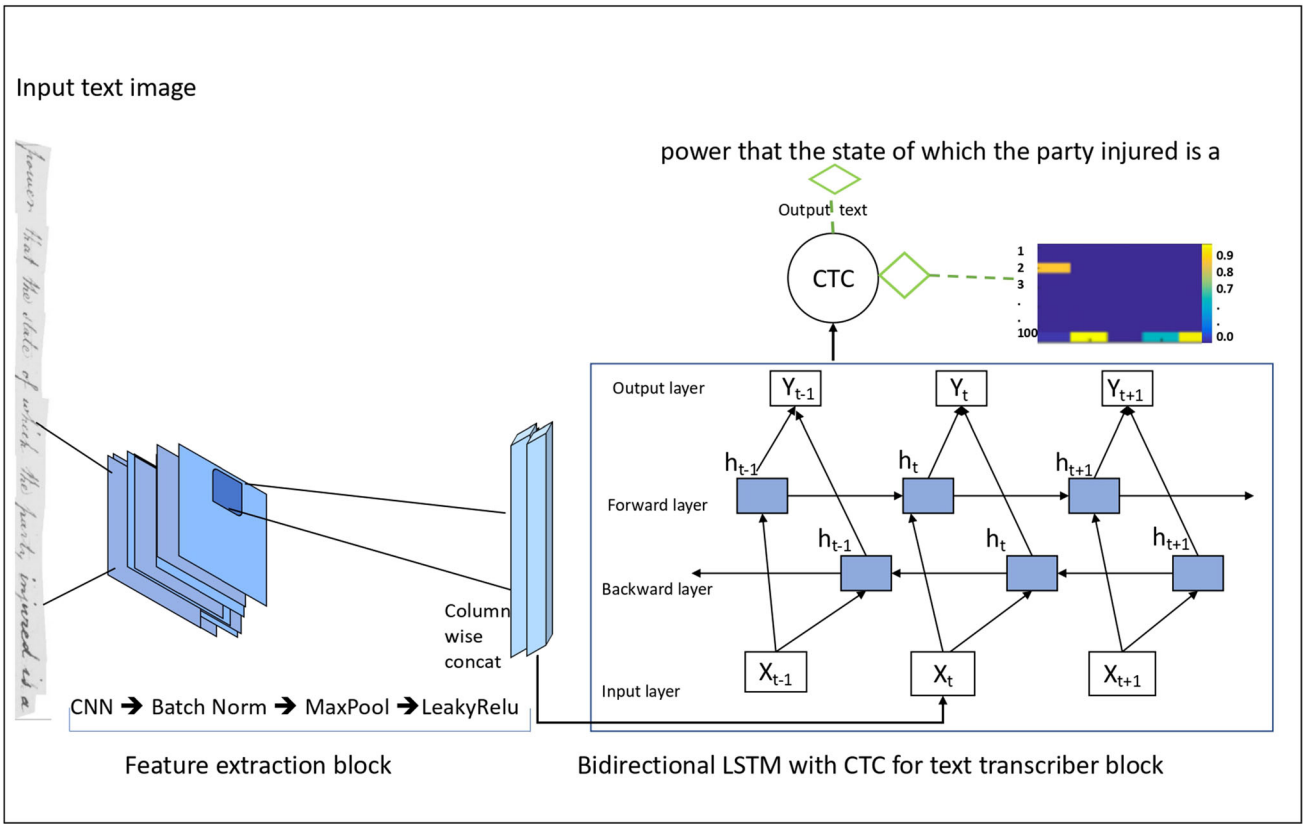


Fig. 1 An overview baseline model: an input line image is fed to five blocks of convolution layers with 3*3 filters each, batch normalization for stability and fast convergence, max-pooling for down-sampling, and leakyRelU activation function for nonlinearity serve as feature extraction block then followed by five blocks of BiLSTM serves as text transcriber with Dense layer, Softmax, and CTC decoder

Figure 1 gives an overview of the baseline HTR model. The proposed based line model takes a text image as input. Then, there are four main parts to the model: the feature extracted from a given input image using the ConvNet method, a self-attention module to enhance the extracted feature representations, the visual representation treated as sequences and outputted their corresponding character probability distributions using BiLSTM method, finally, the last textual transcription obtained from the decoding block using CTC loss/decoding method. We train the baseline model by ensuring that the output sequence of CTC probability distribution is as high as possible. So, in addition to the characters that make up the text, the RNN gets a unique blank character that means "no other character."

We have adopted the self-attention module from [53]. After experimenting with the baseline model, we found that adding a self-attention module to the end of CNN obtained the most robust feature representations and improved model performance. The baseline Handwriting recognition model is trained using CTC loss and further detailed in experimental setup Sect. 4.3.1. The CTC tackles the issue of alignment, as alignment information is not provided for the input data or output transcription since handwriting differs from person to person. The influence of the attention mechanism on the CNN feature extraction method is displayed in Table 4. This method reduced 0.65% on the Bentham dataset while injecting the attention module into the proposed baseline model.

3.2 Preprocessing

The used datasets have a wide range of image sizes. In contrast, the input dimension of models must be fixed for efficient training and a pooling layer. Therefore, images in this paper are downsized to 1024 height and 128 widths with maintaining the aspect ratio. Since we have black and white text-line images, pixels are either 0 or 255. All of the characters and the background in the images are represented as 0 and 255, respectively. So we have added white color padding to the image in order to maintain the aspect ratio after resizing. More specifically, adding the padding where the pixels were short of the desired resolution; otherwise, we first resize along the shorter dimension such that the width is 128 pixels and then, crop the image along the height, such that the height is 1024 pixels. Maintaining a consistent aspect ratio allows our

Convolutional Neural Network to learn more discriminative and compatible features.

Various noises often affect offline handwritten text images scanned by various scanned devices. Also, different writers' handwriting styles (fonts) vary extremely widely. Though our proposed model is an end-to-end HTR system, input images are cleaned using the deslanting method [48] to remove the writing cursive style. Illumination Compensation technique [12] to remove shadows and balance brightness and contrast normalization before feeding to our CNN for feature extraction. In the present system, the following collection of preprocessing operations has been applied (i) illumination compensation techniques, (ii) removal of slant and slope from cursive handwriting text, (iii) normalization of line height to a fixed value (128 units), keeping the aspect ratio unchanged and (iv) a combination of both (i, ii).

The Illumination Compensation Techniques are five processes to balance uneven light distribution. The primary goal of the entire process is to produce content with a high degree of recognition. To eliminate the slanted and skewed text, we extend the most popular technique used as a new normalization approach to solving the cursive text in the word-level handwritten text by removing the slope and the slanted text. The results of those preprocessing experiments are conducted on the Bentham dataset and presented in Sect. 5.1.2.

3.3 Transformer-based model

In this subsection, we explained in detail the proposed HTR convolutional transformer architecture, starting with the feature extraction backbone and then, the transformer component methods.

3.3.1 Feature Extraction using Convolutional Neural Network

We investigate different CNN architectures, specifically the most recent SOTA models, including ResNet [27] and ResNeSt [54] with an attention mechanism for better feature extraction. Tables 6 and 7 report the results on the Bentham dataset as we found out that when the network goes deeper and uses the attention, the better the feature extraction is and hence, the better accuracy. Despite recent advancements in image classification models [31], most downstream applications such as image recognition [27], object detection [45], and semantic segmentation [32] continue to utilize ResNet variations as the backbone network due to their simplicity and modularity. In contrast, we employ a basic and modular Split-Attention block that allows attention to be distributed across feature-map groups. Stacking these Split-Attention blocks on ResNet-style formed a new variant called ResNeSt. ResNeSt is chosen because it preserves the overall ResNet structure in subsequent tasks without charging extra processing costs. ResNeSt beats other SOTA neural networks with comparable model complexity and improves downstream HTR and OCR tasks.

3.3.2 Encoder–Decoder Transformer Network

In this subsection, we describe the main components of the proposed transcriber model architecture:

Encoding feature representation: High-level representations of features are extracted from the line handwritten image ($x_i \in X$) in CNN feature extraction sub-network 3.3.1, where x_i represents a sample of input images as in Fig. 2. The visual representation and sequential order information are encoded. To process the handwritten line images, the input image x_i is first processed by the CNN, which can handle images of any size. We can obtain an intermediate visual feature representation F_v of size ($f = 2048$) as a result of the process. The ResNeSt101 convolutional architecture serves as the backbone of our convolutional architecture. Visual feature representations using standard CNN cannot compact the global representation of the input image; therefore, the attention layer in ResNeSt101 helps extract this meaningful information.

Positional encoding (PE): In Latin scripts, text images are processed sequentially from left to right. While avoiding repetition, Positional Encoding stages before the transformer encoder phase are meant to leverage and encode such crucial information. It also helps the model define the exact location of the next word and character in the text line. For the model to utilize the sequence's order, we employ the positional encoding (PE) of tokens within the sequence. As a result, we incorporate the input embeddings with the PE at the bottom of the decoder, as shown in Fig. 2. The encoded feature vector from the ResNeSt is converted to the same hidden dimension (256) as the transformer encoder before adding the PE. The PE in Eq. (1, 2) are useful in the proposed architecture since they teach our model wherein the sequence is actively focused.

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

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

where pos represents the character at a particular position i, d_{model} is the size of the hidden dimension in the transformer block; both Eq. (1, 2) are used at the input of decoder to add the absolute positional information. We have used the 2D learnable positional encoding for the encoder to get the rich features for character representation.

Decoding visual feature representation: The self-attention is used to further improve the visual features of the

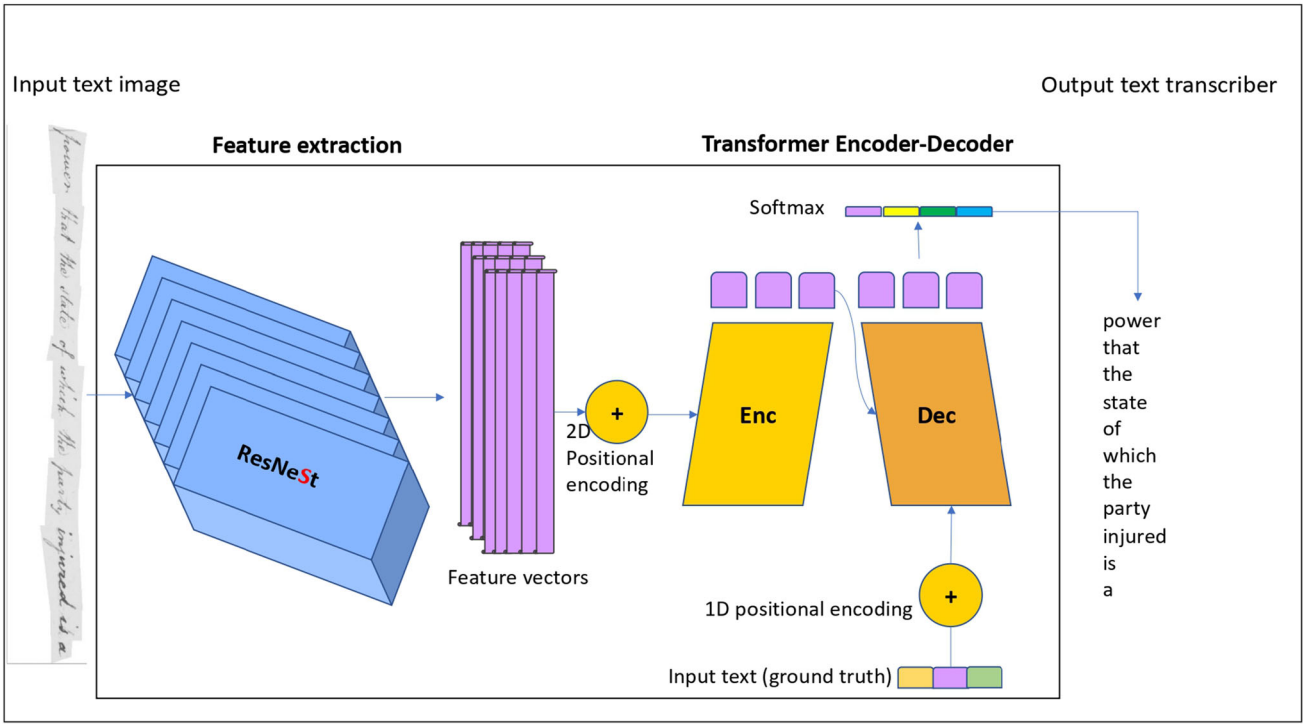


Fig. 2 Proposed model architecture: the Split attention of CNN serves as a feature extractor method where an input line image is fed to the ResNeSt101 backbone that outputs the feature vector, which will be coded using the position encoding technique before providing to the transformer encoder part, and an encoder–decoder transformer network serve as text transcriber method where regularization, batch normalization, and Positional encoder are essential components of the architecture

extracted representations. This module accepts three inputs that are equivalent to $\hat{F}_v$ that represented Q_v, K_v, V_v, for the query, the key, and the value, respectively. Let A_v^i be the correlation information of the attention visual feature obtained by Eq. 3.

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

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, we obtain the high level visual representation from Eq. 3 where $\hat{F}_v = \{A_v^0, A_v^1, , A_v^2, \ldots, A_v^{W-1}\}$. In Fig. 3, we visualize the role of self-attention on the target line image. Due to the short length of that line sequence, we also show the visualization of the padding. The padding shows at the end of the presented image until we reach the maximum length of the input image on the datasets, which is pre-determined as 2048.

Transcriber text decoding : As depicted in Fig. 2 where visual aspects and language-specific information gathered through textual representations are taken care of this component. It outputs the decoded characters and anticipates the next likelihood character following decoded characters sequences. To analyze the line string effectively, we need symbols that do not contain textual information and the numerous characters inspected in the vocabulary size V alphabet. The characters $< BOL >$ and $< EOL >$ are used to indicate the beginning and end of the line sequence, respectively. In contrast, the character $< pad >$ is used to indicate padding, as depicted in Fig. 3. Except for the initial/first character, the transcriptions ($y_i \in Y$) are padded out to a maximum predicted many characters of N. Character level embedding uses a dense layer to convert every character from the input string into an f-dimensional (f_d) vector. Eq. 4 uses the same positional encoding as in Eq (1, 2) where the PE should encode time-steps uniquely. In Fig. 3, we visualize the role of self-attention on the target line image.

$$F_d = \text{Mapping}(y_i) + PE \tag{4}$$

where y_i is the ground-truth text transcription, PE is the positional encoding.

4 Experimental setup

In this section, we examine whether the proposed method (ResNeSt101) is suitable to tackle the problem of the HTR

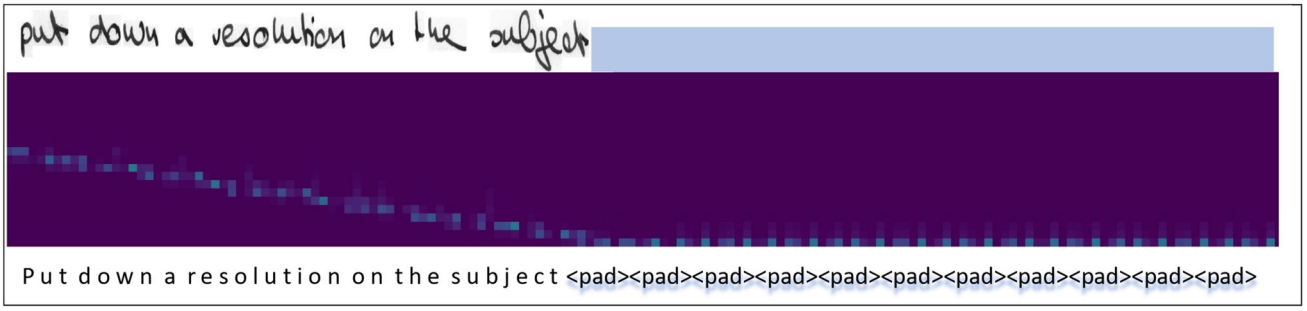


Fig. 3 The impact of self-attention in transformer-decoder network demonstrating the padding for this input line image

Table 1 Descriptions of the partitions on used datasets

Database	Training	Validation	Testing
IAM	6161	900	1,861
RIMES	10193	1,133	778
Washington	325	168	163
Bentham	9195	1415	860

task based on split attention convolutions compared to the other baseline models.

4.1 Datasets

This study examines the HTR system across four datasets: the IAM [33], RIMES [26], Washington [28], and Bentham [10] databases. Table 1 describes the standard partitions used by other researchers on the four datasets to ensure a fair comparison with the current work. Precisely, we can describe the used datasets as follows.

The IAM database has 13353 modern English text lines created by 657 distinct authors. The RIMES database is a French handwriting script collected by 1,300 writers. The George Washington dataset comprises 565 text lines from George Washington's letters authored by two 18th-century writers. Bentham's dataset is English scripts in black-and-white images with distorted writing and dark backgrounds. The text in this collection is about 11,500 lines. A sample of Bentham images is shown in Sect. 5.1.2 along with the corresponding preprocessing methods. Each dataset contains 100 characters, including capital and lowercase letters, numerals, punctuation, special symbols, and white space.

4.2 Performance metrics

Character Error Rate (CER) and Word Error Rate (WER) used to evaluate the proposed HTR model. Both equations (CER 5 and WER 6), in all cases, are the total number of insertion, replacement, and deletion operations required to shift from one sequence to another are based on the Levenshtein edit distance [52]. Mainly, CER is defined as

$$\mathrm{CER} = \frac{1}{|N|} \sum_{(p_i, l_i) \in N} \mathrm{LD}(\hat{y_i}, y_i) \tag{5}$$

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 ith character. In terms of WER, it is defined similarly to CER. Whereas LD($\hat{y}$,y) is computed on word-level that requires to transform one string into another as deletions D word's sum, insertions I and substitutions S by the ground-truth N for that partitions.

$$\mathrm{WER} = \frac{S_{\mathrm{word}} + I_{\mathrm{word}} + D_{\mathrm{word}}}{N_{\mathrm{words}}} \tag{6}$$

4.3 Implementation details

This section describes the implementation of the recurrent baseline model and the proposed transformer-based model.

4.3.1 Baseline recurrent model

This subsection briefly describes the specific implementation details applied to the baseline recurrent model as introduced in the proposed method Sect. 3.1. We stack for both convolution and recurrent layers five blocks each where the convolution layer convolved using 3×3 kernel size. We used zero-padding with stride=1 to maintain the layer's outputs that lead to the exact spatial dimensions as its inputs. A normalization layer (batch norm) is used to help the model converge faster and make it more stable, and a LeakyReLU activation function provides a nonlinearity. Further details, a brief description of the model has depicted in Table 2.

4.3.2 Non-recurrent transformer-based model

The proposed model has a feature size of 2048. The network architecture 3 contains a CNN block for feature extraction, 4 encoder layers, and 4 decoder layers with a hidden dimension of 256. We used a batch size of 16, and ADAM optimizer, and a learning rate of .0006. We adopted the label smoothing technique [37] with Kullback–Leibler divergence cross-entropy

Table 2 Baseline model configuration with the most important hyper-parameters

Layer	Configuration
Input image	1024 × 128 (gray scale)
Convolution layer	16, k: 3 × 3, s: 1, p: same
Batch normalization	
Leaky relu	
Max pooling window	2 × 2, s: 2, p: valid
Convolution Maps	32, k: 3 × 3, s: 1, p: same
Batch norm	
Leaky relu	
Max pooling window	2 × 2, s: 2, p: valid
Dropout	rate: 0.2
Convolution maps	48, k: 3 × 3, s: 1, p: same
Batch Norm	
Leaky Relu	
Max Pooling window	2 × 2, s: 2, p: valid
Dropout	rate: 0.2
Convolution maps	64, k: 3 × 3, s: 1, p: same
Batch Norm	
Leaky Relu	
Self Attention module	
Dropout	rate: 0.2
Convolution Maps	80, k: 3 × 3, s: 1, p: same
Batch Norm	
Leaky Relu	
Self Attention module	
BiLSTM Hidden units1	256
BiLSTM Hidden units2	256
BiLSTM Hidden units3	256
BiLSTM Hidden units4	256
BiLSTM Hidden units5	256
Dropout	rate: 0.5
Decoding	CTC

Table 3 The model architecture

Block	Configuration
ResneSt	
2D CNN	(2048 to 256)
Encoder	
Multi-head attention	×4
Normalization layer	×4
Feed forward layer	×4
Normalization layer	×4
Decoder	
Masked multi-head self-attention	×4
Normalization layer	×4
Multi-head attention	×4
Normalization layer	×4
Feed forward layer	×4
Normalization layer	×4
Dense layer	×1
Softmax layer	×1

loss. Every line is padded to the maximum length of 128 characters using a unique character to the right, as illustrated in Fig. 3. Vocabulary size is 100, which include small/capital letter, punctuation marks, numbers, and other special characters.

4.3.3 Kullback–Leibler divergence loss

We calculated that error regularly during the training optimization process to penalize the error model predictions. Choosing a cost function to estimate the model performance allows for updating the weights to minimize the loss on the next epoch. The loss function used in neural network models must fit the issue framing. The softmax activation function generates a probability distribution of over 100 classes (multi-class classification problem) in our task. In contrast to the Softmax situation, where the categorical cross-entropy loss function is frequently used, the argmax of the predictions generated compares the predicted distribution with the ground truth distribution. Using Kullback–Leibler Divergence (KLD), [36] is an adaptation of the entropy metric standard in information theory. The predictions generated by the final feedforward layer effectively form a probability distribution and can thus be compared to the actual distribution for the sample x in the corresponding training dataset.

KLD computes the gain and loss of the probability distribution between the predicted distribution of the model $P(t)$ and the distribution of the ground-truth $G(t)$. Backpropagation will continue until the model $P(t)$ produces textual transcription equal to or very similar to the ground truth $G(t)$ distribution probability. The model weights and biases will be adjusted in Eq 7 using the ADAM optimizer to achieve the ideal distribution of the prediction probabilities output.

$$\mathrm{KLD}\,(P\|G) = -\sum_{i\in X} P(i) * \log\left(\frac{G(i)}{P(i)}\right) \tag{7}$$

where i represents both a sample of the decoded ground truth of X along with its corresponding encoded image feature representation.

4.3.4 Label smoothing

Over-fitting and overconfidence are common issues when training deep learning models. Some regularization techniques have addressed the over-fitting issue like early stop-

Table 4 The impact of attention mechanism using our baseline model on Bentham dataset with no pre-processing technique

Model	CER(%)
CNN-LSTM	11.75
CNN-atten-LSTM	**11.10**

Bold number shows the influence of attention backbone network against the standard CNN with LSTM decoder, as discussed in Sect. 3.1

ping, weight decay, and dropout. Fortunately, the Label smoothing technique [37] can solve both issues. Equations 8, 9 represent the Label Smoothing that help combine the uniform distribution with updating the one hot coded label vector y that is traditionally used and replacing it with the updated label vector.

$$\hat{y}_i = y_{hot}(1-\alpha) + \alpha/K \tag{8}$$

where (K = 100) is the total number of multi-class categories and y_{hot} represents the embedded ground truth labels.

$$\hat{y}_i = \begin{cases} 1-\alpha, & i = \text{target} \\ \alpha/K, & i \neq \text{target} \end{cases} \tag{9}$$

In this way, the smoothed distribution of the Label is equivalent to adding noise to the actual distribution to avoid the model being too confident about the correct Label. Therefore, the difference between the output values of the predicted positive and negative samples is not so significant to avoid overfitting and improve the model's generalization ability. We experimented with a 0.4 as the value of α. Table 3 presents the attention Transformer model architecture.

5 Results and discussions

As mentioned in Sect. 3.3.1, the proposed architecture is based on ResNeSt backbone for feature extraction. All the models were initialized with Imagenet [18] pre-trained weights. We investigate different SOTA neural network models in both: (1) Feature extractions and representation with and without split attention CNN including ResNeSt Image-net pretrained model. (2) Recurrent and non-recurrent encoder–decoder task using both BiLSTM Sect. 3.1 as baseline model, and Transformer encoder–decoder network as proposed model Sect 3.3.2.

5.1 Preliminary results

In the following Sect. (5.1.1 and 5.1.2), we presented the initial results conducted by our experiments on preprocessing methods and the recurrent baseline model. Sect. 5.1.3 presents the result of different backbones for feature extraction in transformer-based model.

5.1.1 Results on baseline model

Table 4 shows the impact of the attention mechanism on the CNN feature extraction method. The CER/WER per-

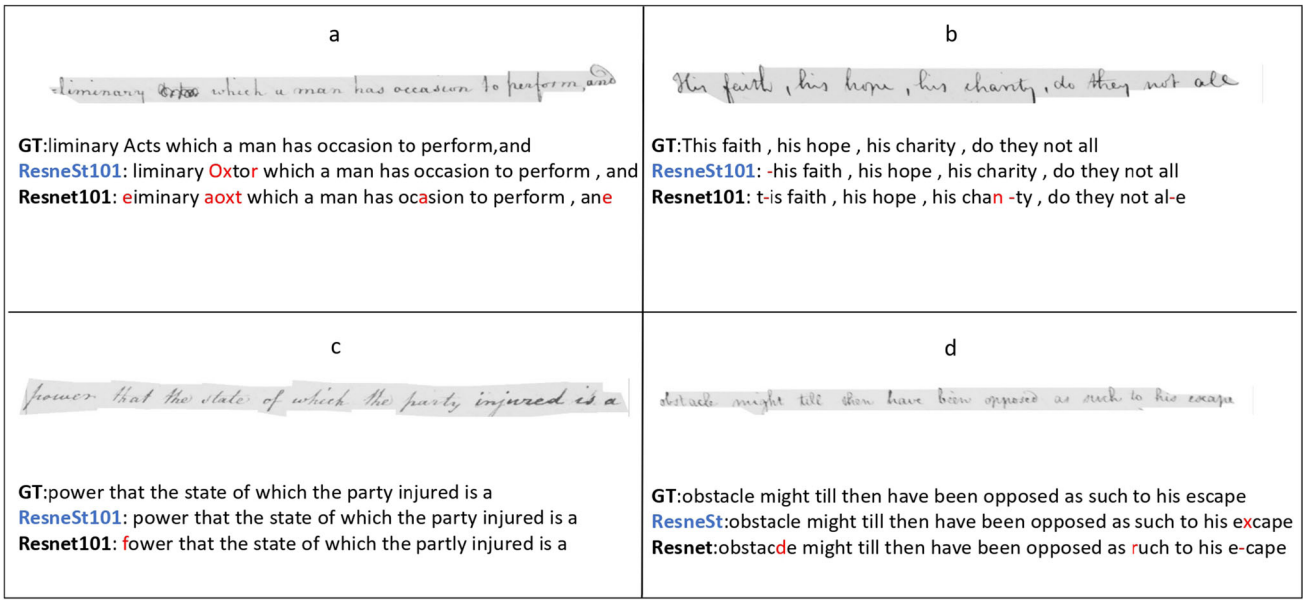


Fig. 4 The impact of the Attention mechanism on the prediction of different input images from the Bentham Dataset. The examples are shown from top to bottom: the input image, corresponding ground truth, and the output prediction with the split attention mechanism is backed by the ResneSt101 CNN. Finally, the output prediction without the attention mechanism is backed by the Resnet101 CNN

Table 5 The impact of preprocessing methods using the proposed baseline attention model on Bentham

Preprocessing	CER(%)
Raw data	11.10
Illumination Compensation (a)	12.03
Deslanted - remove cursive (b)	**10.70**
both (a) and (b)	11.70

Bold number shows the impact of deslanted algorithm on the preprocessing step as discussed in Sect. 3.2

formance is slightly improved using the baseline-attention model on the Bentham dataset. That leads us to choose the ResNeSt split attention backbone in our proposed model, which increased the performance by 2% in favor of the standard ResNet backbone feature extraction method. Further validation supported the importance of the attention mechanism on CNN are the quantitative and qualitative results reported, as, respectively, demonstrated by Table 7 and Fig. 4 in Sect. 5.1.3. Based on that, we confirm that the CNN-attention-based model captures helpful information, hence becoming robust to the earlier HTR task challenges of handwriting variations. Qualitative results provided visualization evidence where the font red denotes the mispredicted characters over the ground truth and support the quantitative findings showing that the suggested technique is more resilient to inter/intra-class handwriting variances.

5.1.2 The impact of preprocessing methods

To elaborate on our choice of the preprocessing techniques discussed in Sect. 3.2, Table 5 demonstrates the CER on four training cases. We choose Bentham line text images to investigate our first hypothesis, the better performance of applying massive or light preprocessing techniques. As a result, we found that only using the removal of cursive handwriting (slant removal text method) leads to the lowest CER. Consequently, we employ only the slant removal method for further experiments and discard the illumination compensation techniques on the proposed datasets.

In Fig. 5, we show the sample from the Bentham dataset after reducing the noise, enhancing the lightening, recovering the damaged strokes, and correcting the geometry distortions (correcting the text skew).

As presented in Table 5, we obtain the lowest error rate when only removing the cursive handwriting. As discussed in 3.2, the preprocessing technique employs the de-slanted algorithm to correct the slanted and skewed handwritten text. To elaborate on our choice of the preprocessing techniques discussed in Sect. 3.2, Table 5 demonstrates the CER on four training cases. We choose Bentham line text images to investigate our first hypothesis, the better performance of applying massive or light preprocessing techniques. As a result, we found that only using the removal of cursive handwriting (slant removal text method) leads to the lowest CER. Consequently, we employ only the slant removal method for further experiments and discard the illumination compensation techniques on the proposed datasets. For further experiments, we utilized the removal of cursive handwriting from the images.

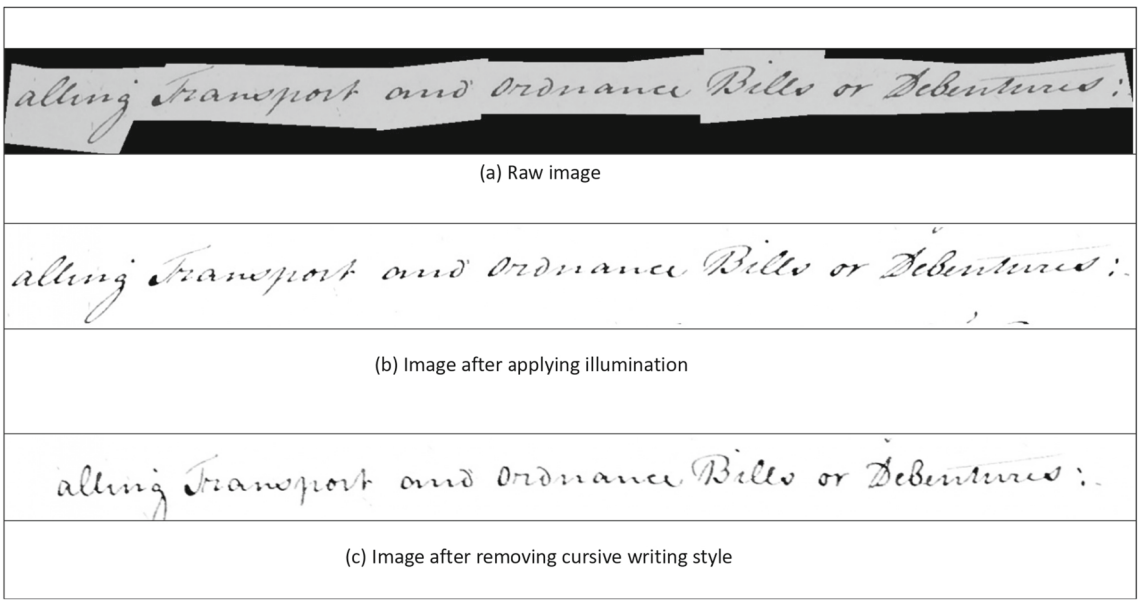


Fig. 5 A sample input image from the Bentham dataset. Top: original image. Middle: an image after applying illumination compensation that cleans any background noise. Bottom: an input image after applying desalinated text removal as pre-processing techniques

Table 6 The impact of deeper layers on Bentham with different backbone feature extraction methods ResNet50 vs ResNet101

Model	CER(%)	WER(%)
ResNet50	7.56	19.01
ResNet101	**6.11**	**14.88**

Bold numbers show the best performance with deeper convolutional network (Resnet50 bs Resnet101) as discussed on Sect. 5.1.3

Table 7 The impact of ResNeSt101 attention mechanism on Bentham dataset as feature extraction

Model	CER(%)	WER(%)
ResNet101	6.11	14.88
ResNeSt101	**4.90**	**12.45**

Bold numbers show the best accuracy with attention backbone (ResNeSt) over standard convolutional (Resnet) backbone network, as discussed in Sect. 5.1.3

5.1.3 Results on Transformer-based model

After the findings that attention mechanism helps in better feature extraction, we also wanted to experiment with the deeper networks. Hence, we first train Transformer model with ResNet CNNs. Table 6 showcases the results based on ResNets experiment.

Since we obtained better results while using deep layers, we experimented on Bentham dataset to compare the ResNet101 with ResNeSt101 for backbone feature extraction; as anticipated, the latter model provides the more robust feature representation of handwriting images shown in Table 7. Therefore, we considered the ResNeSt as our backbone in the architecture for further experiments, as shown in Fig. 2 and Table 3. The Resnet and ResneSt models perform competitively on the Bentham dataset in the considered standard-CNN and the split attention-CNN approaches. Compared to the Standard CNN-baseline model, the proposed model further decreases both the CER and the WER which can also be observed from the qualitative results reported in Fig. 4 where the reddish color indicates the mispredicted characters over the ground-truth. The qualitative result shows that the standard backend CNN method has twice CER than the backend CNN with attention ResNeSt. This suggests that, by capturing the more contextual feature representation, the split attention CNN model makes fewer character-level errors and word-level errors with more than 2%. The result of the Bentham historical dataset demonstrates the improvement of model performance when using the attention mechanism.

Further to the importance of the attention mechanism in the feature extraction block, Table 6 presents two experiments on the Bentham dataset. We discovered that the deeper the network with attention mechanism gets, the better the feature extraction, which ultimately leads to improved accuracy.

5.2 Ablation study

From our findings on previous section, we reported that the network with attention mechanism, the better the feature representation extractions obtained with respect to the best preprocessing choice made by our experiments. We built upon those findings by further improving the SOTA deep learning method for feature extraction and encoder–decoder text transcriber as described in Sect. 3.3.

In ablation research, characteristics from a model are often removed to enhance the model efficiency. The influence on performance is assessed to witness whether or not the model can withstand the removal of specific approaches. This section intensely detailed the impact of transfer learning (TL), data augmentation (DA), and their possible combinations.

5.2.1 An impact of transfer learning

The challenge of training a CNN from scratch is due to different factors, particularly CNN data-hungry and time-consuming. Therefore, TL and DA can solve the HTR task with the shortage of labeled datasets. The Bentham database is excluded from comparing DA with TL since the Bentham dataset plays the source database role in the TL approach, IAM, RIMES, and Washington target datasets. The performance findings of both CER/WER were computed in two scenarios:

- Ensemble approach where we have the average weights of two models trained on 100 and 200 epochs, respectively.
- Best loss where the model records the best validation loss.

Table 8 demonstrates the effect of transfer learning where the source dataset was Bentham. Initially, we trained our model to predict the handwritten text on the Bentham dataset; then, we took the model with the best validation loss for TL on the target datasets, namely IAM, RIMES, and Washington. We also investigated which blocks in ResNeSt should be frozen for best performance. We found out that it is best to train all the blocks. The lowest achieved CER/WER error rates are highlighted in the same Table. It is essential to ensure a reliable comparison with current methods; we used the exact training and testing partitions on a public dataset. The training, validation, and testing procedures used the corresponding set sizes for each dataset are given in partition Table 1.

Table 8 The impact of Transfer Learning where five different scenarios were held on Bentham dataset as the source database concerning the mentioned targeted databases

Feezed blocks	Technique	CER(%)			WER(%)		
		IAM	Rimes	Washington	IAM	Rimes	Washington
All Free	Ensemble	**7.48**	5.35	**6.94**	**21.61**	**11.53**	**19.55**
	Best loss	7.49	**5.26**	7.33	21.60	11.54	20.38
Freeze 1	Ensemble	7.77	5.67	7.36	22.22	11.96	20.59
	Best loss	7.91	5.69	7.17	22.37	11.89	20.19
Freeze 1,2	Ensemble	9.16	6.92	8.58	25.47	12.55	23.44
	Best loss	9.31	6.94	8.88	25.58	12.65	24.08
Freeze 1,2,3	Ensemble	17.18	14.51	21.17	41.11	22.11	47.94
	Best loss	19.61	14.93	20.90	45.10	23.20	48.49
Freeze 1,2,3,4	Ensemble	21.10	17.85	19.14	47.49	25.88	45.52
	Best loss	21.14	17.91	20.59	47.96	25.45	48.81

Bold numbers show the impact of transfer learning from Bentham dataset (large dataset) with 5 different scenarios - where the best findings was with releasing all layers (free) as discussed in Sect. 5.2.1

5.2.2 An impact of both data augmentation and transfer learning

We further analyze the CER/WER performance of both TL and DA techniques when applied to the handwriting system. We followed the same approach as [2]. As they proposed, the alternate techniques (TL and DA) combination has several possible structures. In the first approach, as shown in Table 9, we use DA to learn from the source database (Bentham) and retrain this pre-trained model on the target datasets (IAM, Rimes, Washington).

- The model is trained from scratch using a source dataset augmented with data.
- The model is retrained using target datasets after the data augmentation technique has been applied to the source dataset

this is referred to as the DA-TL-DA paradigm.

On the other hand, in the second proposal, we used the DA technique but did not apply it to the target datasets:

- The model is trained from scratch using a source Bentham dataset using the DA technique.
- the model is retrained without using DA on the target datasets.

To be thorough, in Table 9, we report the findings on the proposed architecture with all block layers unfrozen. The baseline model is the CNN-Transformer model with no techniques applied to it. The DA-TL and DA-TL-DA techniques use the Bentham database to train the model from scratch. The model is then calibrated using data from the IAM, Rimes, and Washington databases, with/without augmentation on the target datasets. As imposed to [2], we can conclude that applying DA over the target datasets after TL is applied, i.e., the DA-TL-DA paradigm, does improve the CER and WER performance. In the case of the Rimes dataset, the DA-TL-DA approach did not perform well; one possible reason for this poor performance could be that the French language on the target dataset is slightly different from the source Bentham English dataset. We acknowledged that applying DA on target datasets following TL is advantageous. The DA-TL-DA paradigm is beneficial when the target source has only a few textually labeled lines, as with the Washington database. After considering the arguments above and Tables 8 and 9, it can be concluded that the DA-TL-DA technique is reliable. The starting point is relatively good when fine-tuning a ResNeSt trained on a similar task, such as the Bentham dataset as an extensive database of HTR samples. Without additional training on the target datasets, we demonstrate that the model can provide good generalization for the DA as in the Rimes dataset and DA+TL-DA as in the rest target datasets. The proposed model is then trained against the target databases containing only a few input samples, as in the Washington database, representing only a tiny portion of the training set.

5.3 Comparison with the state-of-the-art on benchmark datasets

In Table 10, we present a comprehensive performance comparison with the state-of-the-art. Different approaches have been compared to our work depending on whether they required a pre-segmentation task, known as segmentation-based, or not, known as segmentation-free, and whether they required a language model (lexicon-based) or not (lexicon-free). Some techniques are based on recurrent neural networks, typically LSTMs with a Connectionist Temporal Classification (CTC) loss function or Transformer encoder–decoder sequence-to-sequence architectures.

Table 9 The impact of Transfer Learning and data augmentation where five different scenarios were held on Bentham dataset as the source database concerning the mentioned targeted databases

Method	Technique	CER(%)			WER(%)		
		IAM	Rimes	Washington	IAM	Rimes	Washington
Baseline	Ensemble	8.71	4.51	72.59	25.31	12.82	90.68
	Best loss	8.97	4.59	75.79	25.66	12.92	91.39
+TL	Ensemble	7.48	5.35	6.94	21.61	11.53	19.55
	Best loss	7.49	5.26	7.33	21.60	11.45	20.38
+ DA	Ensemble	7.85	**4.24**	70.81	21.46	11.04	88.85
	Best loss	7.77	4.25	73.68	21.46	**10.03**	89.78
+ DA+TL	Ensemble	6.98	4.86	5.76	19.50	13.85	16.36
	Best loss	7.10	4.66	6.01	19.90	13.41	16.38
+ DA+TL+DA	Ensemble	**6.66**	5.53	4.84	**18.97**	12.48	13.75
	Best loss	6.70	5.42	**4.74**	18.90	11.98	**13.23**

Bold numbers show the best performance on our model while carrying out both transfer learning and data augmentation as discussed in Sect. 5.2.2

Table 10 Results of text recognition on four benchmark databases: IAM, Rimes, Washington, Bentham datasets

Model	CER(%)				WER(%)			
	IAM	Rimes	Wash.	Ben.	IAM	Rimes	Wash.	Ben.
Chen et al. [14]	11.15	8.29	–	–	34.55	30.54	–	–
Pham et al. [39]	13.92	8.62	–	–	31.48	27.01	–	–
Wigington, et al. [49]	6.07	3.09	17.50	–	19.07	11.29	65.20	–
Aradillas et al. [2]	**5.90**	**2.70**	18.70	–	20.30	10.70	69.20	–
Bluche et al. [6]	3.20	1.90	–	–	10.50	7.90	–	–
Kang et al. [30]	4.67	–	–	–	15.45	–	–	–
Flor et al. [17]	8.52	6.78	–	8.49	30.58	29.75	–	24.04
Puigcerver et al. [43]	9.32	8.56	–	8.16	32.13	31.29	–	24.43
Cascianelli et al. [9]	6.8	4.0	–	–	24.7	13.7	–	–
Ours	6.66	4.24	**4.74**	**4.90**	**18.97**	**10.03**	**13.23**	**12.45**

Bold numbers present the best model performance on different datasets with comparison to other related models as discussed in Sect. 5.3

Our results are compared against the nine related studies to demonstrate the improvement and evaluate the accuracy of the proposed model. These approaches used deep learning methods based on CNN and LSTM-CTC or Transformer encoder–decoder, where some variant of the LSTM is used. Some studies like using DA, synthetic datasets in the target datasets, and Lexicon Model (LM). We track the percentage of characters and words incorrectly identified as CER and WER, respectively. The proposed model is tested on four benchmark datasets using their standard partitions as described in our experiments in Sect. 4.1, to ensure a fair comparison with the state-of-the-art methods in Table 10. Additional experimental protocols used in the existing studies to further compare our work are: (1) recurrence or nonrecurrence approaches, (2) employed DA-TL or not, (3) lexicon-based or lexicon-free model, and (4) using synthetic dataset or not. Though Chen et al. [14] attempt to provide a multi-HTR systems-based recurrent network, their model poorly performed on IAM and Rimes compared to our finding. Although Pham et al. [39] proposed and empowered their model with Dropout and a constrained language and lexicon-based model, their model performance remains farther than ours. Likewise, in our proposed model, Wigington et al. [49] applied DA to their studies, and we got close CER/WER in IAM and Rimes datasets. However, in the Washington dataset, we outperform their approach with more than 12%. Similarly, Aradillas et al. [2] outperform our (DA+TL+DA) paradigm approach, with a slight value of less than 1% CER on IAM and around 2% CER on the RIMES dataset. However, we obtain better WER performance on those datasets as well as on the Washington dataset. Bluche et al. [6] and Kang et al. [30] outperform the proposed model on IAM and Rimes with some constraints on their models like generating synthetic datasets and applying language model. However, the improvement is significantly noticed using the transformer as the decoder part; unlike our work, Bluche and Kang boost their performance with the help of generating a synthetic dataset and using the lexicon-based model. Finally, our proposed model marginally outperforms the performance of the work of Flor et al. [17], Puigcerver et al. [43] and Cas-

Table 11 The impact of CLIP model pre-training with Augmentation on IAM dataset

Method	Technique	IAM	
		CER(%)	WER(%)
No Pre-training	Ensemble	7.85	21.46
	Best loss	7.77	21.30
Pre-training	Ensemble	7.65	21.36
	Best loss	**7.59**	**21.23**

cianelli et al. [9]. However, Puigcerver used DA and synthetic datasets to empower their performance model. In contrast, the work of Cascianelli employed deformable convolutions that resulted in competitive CER/WER in IAM and Rimes datasets.

During the evaluation process, we also try to use Clip Pre-training on the IAM dataset. Since we aimed to discover the effect of CLIP [44] pre-training the backbone CNN model on the performance of HTR's CER since the CLIP is exceptionally well-known for its capabilities these days. We decided to use the Clip-pretraining on the ResNeSt model. The approach is similar to what is described in the CLIP model. The ground truth labels are fed to the text encoder sub-encoder model and output a list of their corresponding textual embedding, called mathematical space. On the other hand, the equivalent input images fed to the sub-encoder model output its relative feature representations that serve as input for the Transformer encoder.

After pre-training the CNN model on the IAM dataset, we used the same weights to train the CNN-Transformer model on the HTR task. Table 11 shows the impact of clip training on the CER for the IAM dataset. DA is done on both of the experiments. We can conclude that pre-training the CNN model helps improve the model's performance.

6 Conclusion

This paper presents a transformer-based and lexicon-free approach for HTR. To the best of our knowledge, this is the first approach to the HTR task that utilizes a joint attention mechanism on both feature extraction (CNN) and encoder–decoder text transcriber (transformer) networks. We examined different preprocessing methods before feeding data to the purposed model; we found that removing only the slanted text technique improves performance. We conducted a rigorous analysis and evaluation of several experiments with (TL) and (DA) paradigms, demonstrating DA+TL+DA paradigm opted to be the proposed and optimal design for the HTR task. Our findings on the reported results show that our method can achieve the best possible results (SOTA) with neither synthetic data nor a lexicon model. Also, our proposed model can deal with small-shot training datasets such as the Washington dataset, extending its relevance to real-world use cases. Transformers are excellent at integrating visual and language-specific knowledge since they are character-based rather than vocabulary-based.

More specifically, the HTR problem is a generic text recognition task. Therefore, we provided an end-to-end neural network model trained on variable-sized line images with their corresponding line-level transcriptions. To summarize our contributions on the following lines:

- We studied the impact of the attention mechanism on CNN-backbone feature extraction
- We examined the influences of the prior utilizing different preprocessing techniques.
- We devised a unified deep learning framework from SOTA dual attention mechanism networks
- We comprehensively investigated the effect of TL and DA.
- Our proposed model with lexicon-free and no synthetic data outperforms the performance of the cited SOTA models with the same constraints.

We completed thorough evaluations on four public benchmark datasets for handwriting text recognition. We achieved SOTA performance utilizing the identical architecture and the minimum hyper-parameter changes. That makes our model more robust, universal, and straightforward for any new text recognition challenge. Extensive experiments are performed on four public datasets, and both the source code and all of the pre-trained models will be available on our GitHub.

In the future, we aim to improve the architectural design by incorporating (TL) and (DA) with the CLIP model to decrease the CER/WER further. In addition, synthesizing good enough handwriting text lines will help the model train on a substantial amount of data that improve the model performance and addresses overfitting. The closed vocabulary (LM) should incorporate the DA-TL-DA and CLIP techniques to improve CER/WER.

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

References

1. Aberdam, A., Litman, R., Tsiper, S., Anschel, O., Slossberg, R., Mazor, S., Manmatha, R., Perona, P.: Sequence-to-sequence contrastive learning for text recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 15302–15312 (2021)
2. Aradillas, J.C., Murillo-Fuentes, J.J., Olmos, P.M.: Boosting offline handwritten text recognition in historical documents with few labeled lines. IEEE Access **9**, 76674–76688 (2021)

3. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate (2014). arXiv preprint arXiv:1409.0473
4. Bhunia, A.K., Khan, S., Cholakkal, H., Anwer, R.M., Khan, F.S., Shah, M.: Handwriting transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 1086–1094 (2021)
5. Bianne-Bernard, A.L., Menasri, F., Mohamad, R.A.H., Mokbel, C., Kermorvant, C., Likforman-Sulem, L.: Dynamic and contextual information in hmm modeling for handwritten word recognition. IEEE Transact. pattern. Anal. Mach. Intell. **33**(10), 2066–2080 (2011)
6. Bluche, T., Messina, R.: Gated convolutional recurrent neural networks for multilingual handwriting recognition. In: 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), vol. 1, pp. 646–651. IEEE (2017)
7. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. Adv. Neural Inform. Process. Syst. **33**, 1877–1901 (2020)
8. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: European Conference on Computer Vision, pp. 213–229. Springer (2020)
9. Cascianelli, S., Cornia, M., Baraldi, L., Cucchiara, R.: Boosting modern and historical handwritten text recognition with deformable convolutions. International Journal on Document Analysis and Recognition (IJDAR) 1–11 (2022)
10. Causer, T., Wallace, V.: Building a volunteer community: results and findings from Transcribe Bentham. Digit. Humanit. Q. **6**(2) (2012)
11. Chammas, E., Mokbel, C., Likforman-Sulem, L.: Handwriting recognition of historical documents with few labeled data. In: 2018 13th IAPR International Workshop on Document Analysis Systems (DAS), pp. 43–48. IEEE (2018)
12. Chen, K.N., Chen, C.H., Chang, C.C.: Efficient illumination compensation techniques for text images. Digital Signal Process. **22**(5), 726–733 (2012)
13. Chen, X., Mishra, N., Rohaninejad, M., Abbeel, P.: Pixelsnail: An improved autoregressive generative model. In: International Conference on Machine Learning, pp. 864–872. PMLR (2018)
14. Chen, Z., Wu, Y., Yin, F., Liu, C.L.: Simultaneous script identification and handwriting recognition via multi-task learning of recurrent neural networks. In: 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), vol. 1, pp. 525–530. IEEE (2017)
15. Cheng, J., Dong, L., Lapata, M.: Long short-term memory-networks for machine reading (2016). arXiv preprint arXiv:1601.06733
16. Cui, Z., Ke, R., Pu, Z., Wang, Y.: Deep bidirectional and unidirectional lstm recurrent neural network for network-wide traffic speed prediction (2018). arXiv preprint arXiv:1801.02143
17. de Sousa Neto, A.F., Bezerra, B.L.D., Toselli, A.H., Lima, E.B.: Htr-flor++ a handwritten text recognition system based on a pipeline of optical and language models. In: Proceedings of the ACM Symposium on Document Engineering 2020, pp. 1–4 (2020)
18. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255. IEEE (2009)
19. 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
20. Dong, L., Xu, S., Xu, B.: Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5884–5888. IEEE (2018)
21. Espana-Boquera, S., Castro-Bleda, M.J., Gorbe-Moya, J., Zamora-Martinez, F.: Improving offline handwritten text recognition with hybrid hmm/ann models. IEEE Transact. Pattern Anal. Mach. Intell. **33**(4), 767–779 (2010)
22. 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, pp. 369–376 (2006)
23. Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., Schmidhuber, J.: A novel connectionist system for unconstrained handwriting recognition. IEEE Transact. Pattern Anal. Mach. Intell. **31**(5), 855–868 (2008)
24. Graves, A., Schmidhuber, J.: Offline handwriting recognition with multidimensional recurrent neural networks. In: Koller, D., Schuurmans, D., Bengio, Y., Bottou, L. (eds.) Advances in Neural Information Processing Systems, vol. 21, pp. 545–552. Curran Associates, Inc. (2008). https://proceedings.neurips.cc/paper/2008/file/66368270ffd51418ec58bd793f2d9b1b-Paper.pdf
25. Gregor, K., Danihelka, I., Graves, A., Rezende, D., Wierstra, D.: Draw: A recurrent neural network for image generation. In: International Conference on Machine Learning, pp. 1462–1471. PMLR (2015)
26. Grosicki, E., Carré, M., Brodin, J.M., Geoffrois, E.: Results of the rimes evaluation campaign for handwritten mail processing. In: 2009 10th International Conference on Document Analysis and Recognition, pp. 941–945. IEEE (2009)
27. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778 (2016)
28. Kane, S., Lehman, A., Partridge, E.: Indexing george washington's handwritten manuscripts. Center for Intelligent Information Retrieval, Computer Science Department, University of Massachusetts, Amherst, MA 1003 (2001)
29. Kang, D., Lv, Y., Chen, Y.Y.: Short-term traffic flow prediction with lstm recurrent neural network. In: 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), pp. 1–6. IEEE (2017)
30. Kang, L., Riba, P., Rusiñol, M., Fornés, A., Villegas, M.: Pay attention to what you read: Non-recurrent handwritten text-line recognition (2020). arXiv preprint arXiv:2005.13044
31. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Commun. ACM **60**(6), 84–90 (2017)
32. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440 (2015)
33. Marti, U.V., Bunke, H.: The iam-database: an english sentence database for offline handwriting recognition. Inter. J. Doc. Anal. Recognit. **5**(1), 39–46 (2002)
34. Mermelstein, P., Eyden, M.: A system for automatic recognition of handwritten words. In: Proceedings of the October 27-29, 1964, fall joint computer conference, part I, pp. 333–342 (1964)
35. 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), pp. 1286–1293. IEEE (2019)
36. Moreno, P., Ho, P., Vasconcelos, N.: A Kullback-Leibler divergence based kernel for SVM classification in multimedia applications. Adv. Neural Inf. Process. Syst. **16** (2003)
37. Müller, R., Kornblith, S., Hinton, G.E.: When does label smoothing help? In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F. (eds.) Advances in Neural Information Processing Systems, vol. 32. Curran Associates,

Inc. (2019). https://proceedings.neurips.cc/paper/2019/file/f1748d6b0fd9d439f71450117eba2725-Paper.pdf
38. Parikh, A.P., Täckström, O., Das, D., Uszkoreit, J.: A decomposable attention model for natural language inference (2016). arXiv preprint arXiv:1606.01933
39. Pham, V., Bluche, T., Kermorvant, C., Louradour, J.: Dropout improves recurrent neural networks for handwriting recognition. In: 2014 14th International Conference on Frontiers in Handwriting Recognition, pp. 285–290. IEEE (2014)
40. Plizzari, C., Cannici, M., Matteucci, M.: Skeleton-based action recognition via spatial and temporal transformer networks. Comput. Vision Image Understanding **208**, 103219 (2021)
41. Plötz, T., Fink, G.A.: Markov models for offline handwriting recognition: a survey. Inter. J. Doc. Anal. Recognit. (IJDAR) **12**(4), 269–298 (2009)
42. Poznanski, A., Wolf, L.: Cnn-n-gram for handwriting word recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2305–2314 (2016)
43. 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, pp. 67–72. IEEE (2017)
44. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In:International Conference on Machine Learning, pp. 8748–8763. PMLR (2021)
45. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016)
46. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Transact. Pattern Anal. Mach. Intell. **39**(11), 2298–2304 (2016)
47. Sueiras, J., Ruiz, V., Sanchez, A., Velez, J.F.: Offline continuous handwriting recognition using sequence to sequence neural networks. Neurocomputing **289**, 119–128 (2018)
48. Vinciarelli, A., Luettin, J.: A new normalization technique for cursive handwritten words. Pattern Recognit. Lett. **22**(9), 1043–1050 (2001)
49. 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 (ICDAR), vol. 1, pp. 639–645. IEEE (2017)
50. Yadav, S.S., Jadhav, S.M.: Deep convolutional neural network based medical image classification for disease diagnosis. J. Big Data **6**(1), 1–18 (2019)
51. Yang, Z., He, X., Gao, J., Deng, L., Smola, A.: Stacked attention networks for image question answering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 21–29 (2016)
52. Yujian, L., Bo, L.: A normalized levenshtein distance metric. IEEE Transact. Pattern Anal. Mach Intell. **29**(6), 1091–1095 (2007)
53. Zhang, H., Goodfellow, I., Metaxas, D., Odena, A.: Self-attention generative adversarial networks. In: International Conference on Machine Learning, pp. 7354–7363. PMLR (2019)
54. 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
55. Zhang, Y., Nie, S., Liu, W., Xu, X., Zhang, D., Shen, H.T.: Sequence-to-sequence domain adaptation network for robust text image recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2740–2749 (2019)

Array 19 (2023) 100300

Contents lists available at ScienceDirect

Array

journal homepage: www.elsevier.com/locate/array

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.

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).

https://doi.org/10.1016/j.array.2023.100300
Received 7 May 2023; Received in revised form 12 June 2023; Accepted 14 June 2023
Available online 20 June 2023

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, Coquenet 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.

Coquenet 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

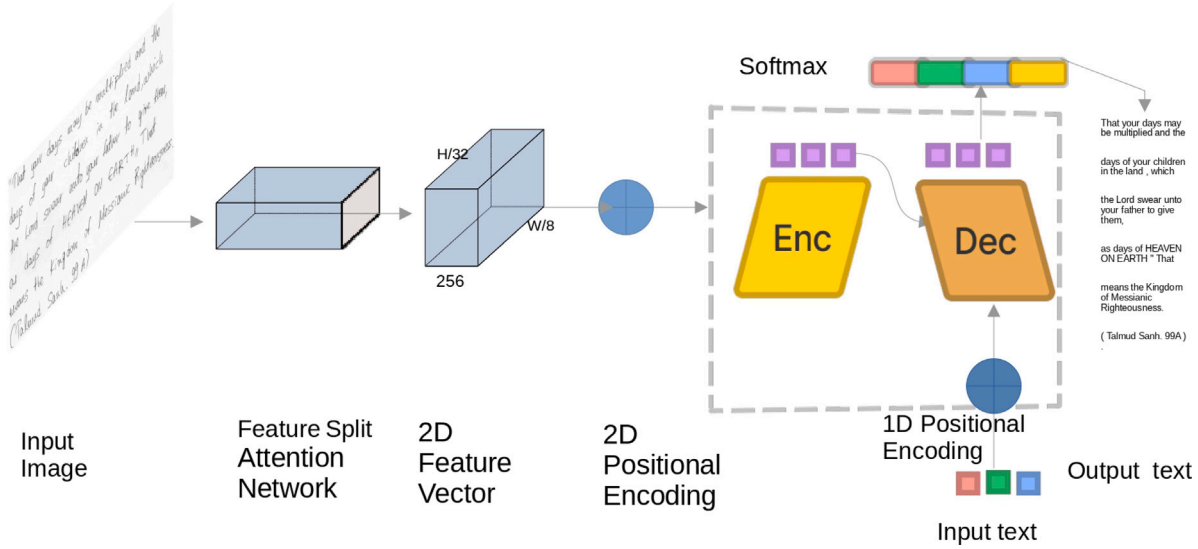


Fig. 1. Network architecture overview: (left) A paragraph input image is fed to a CNN split attention that serves as the backbone for the feature extraction method. (right) A 2D positional encoded technique encodes the CNN output feature vector before feeding it to the encoder self-attention transformer 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.

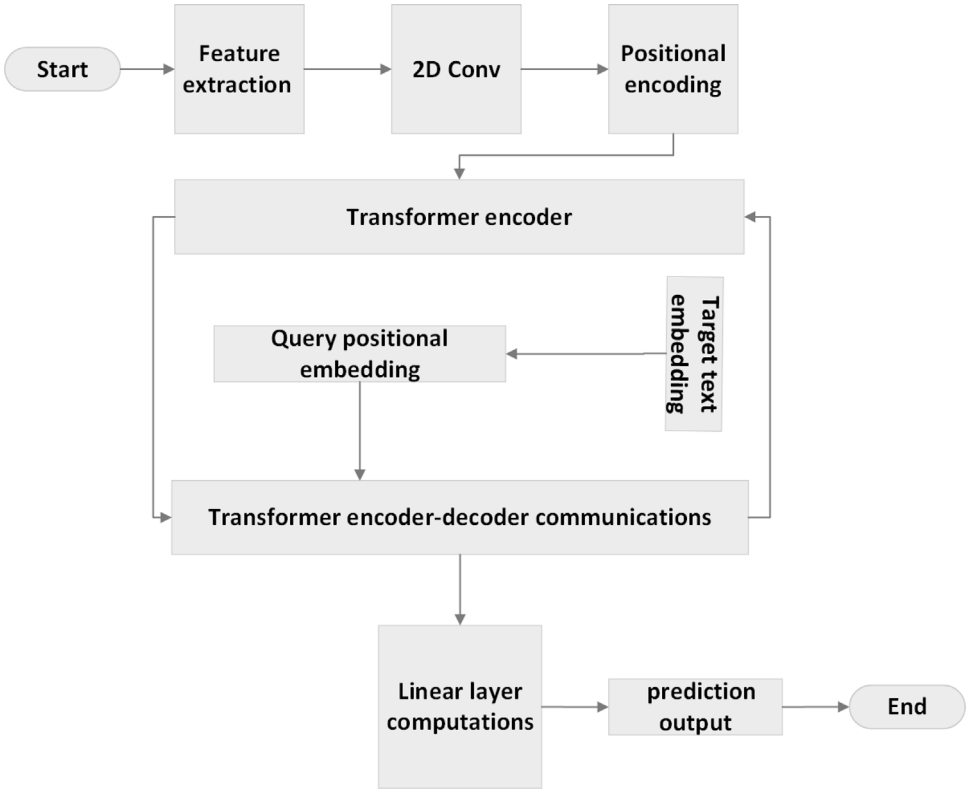


Fig. 2. This flowchart illustrates the sequential steps involved in the HPR process.

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.

$$PE_{(pos,2i)} = sin(\frac{pos}{10\,000^{\frac{2i}{d_{model}}}}) \tag{1}$$

$$PE_{(pos,2i+1)} = cos(\frac{pos}{10\,000^{\frac{2i+1}{d_{model}}}}) \tag{2}$$

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),

$$A_v^i = softmax(\frac{q_v^i.K_v}{\sqrt{f}})V_v \tag{3}$$

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 $\hat{F}_v = \{A_v^0, A_v^1, A_v^2, \ldots, A_v^{W-1}\}$, 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 earn 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.

$$F_t = Mapping(y_t) + PE \tag{4}$$

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,

$$A_v^i t = softmax(\frac{q_t^i.K_v}{\sqrt{f}})V_v \tag{5}$$

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 $\hat{F}_{vt} = \{A_{vt}^0, A_{vt}^1, A_{vt}^2, \ldots, A_{vt}^{L-1}\}$, 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).

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

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 yt, 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.

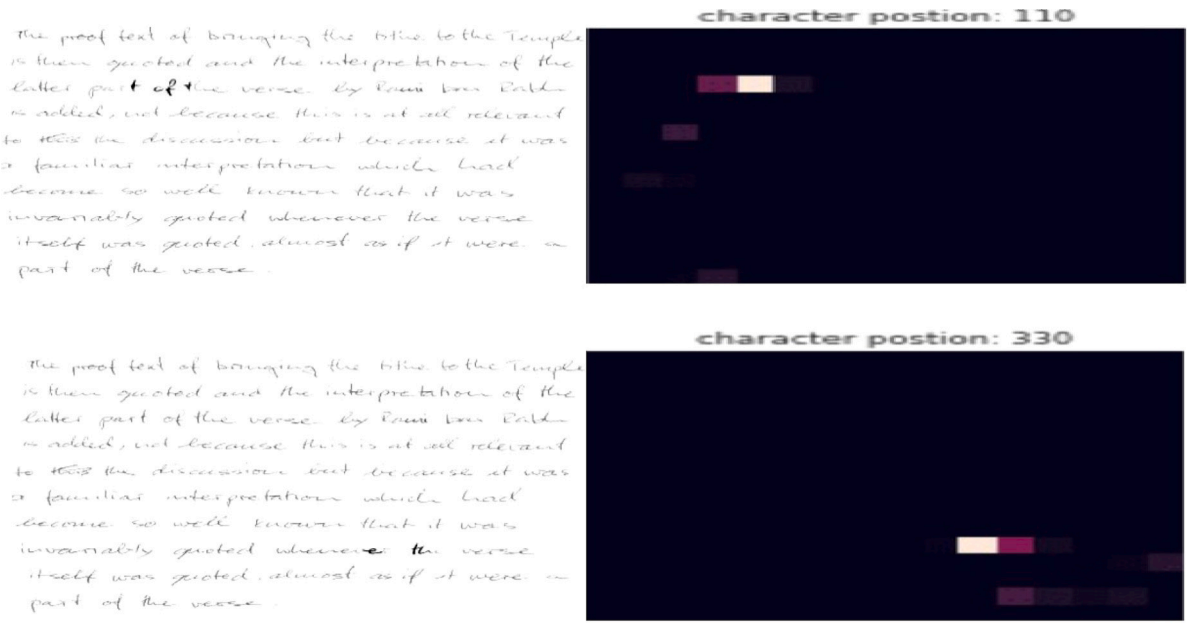


Fig. 3. Paragraph heat-map at decoder layer multi-head self-attention: randomly picked at 110 and 330 position character sequences.

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 ResneSt-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 '¬' 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).

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

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 ith 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.

$$WER = \frac{S_{word} + I_{word} + D_{word}}{N_{words}} \tag{8}$$

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

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

Algorithm 1 shows the training procedures; we first extract the features from the input image: F = backbone(I), followed by reducing the feature dimensions using the convolution layer: H = 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 = transformer_decoder(H', T', M_T, M_P) to predict the output character probabilities: Y = 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 mispredictions. 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 $P(t)$ and the ground truth distribution $G(t)$, we used KLD. The backpropagation operation was performed iteratively until the predicted distribution $P(t)$ yielded a textual transcription that closely matched or was identical to the ground truth distribution $G(t)$. 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 $P(t)$ and $G(t)$ 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.

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

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.

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

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.

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

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.

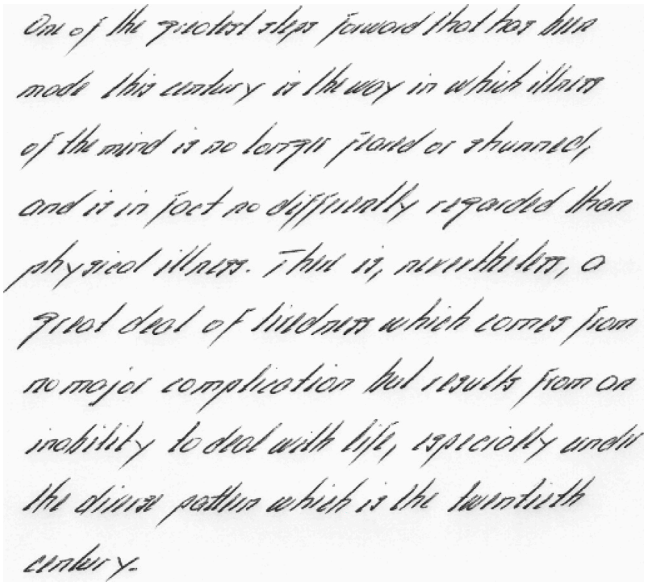


Fig. 4. A sample image of generated synthetic datasets from the IAM line dataset.

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, ResneSt50 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 resneSt-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.

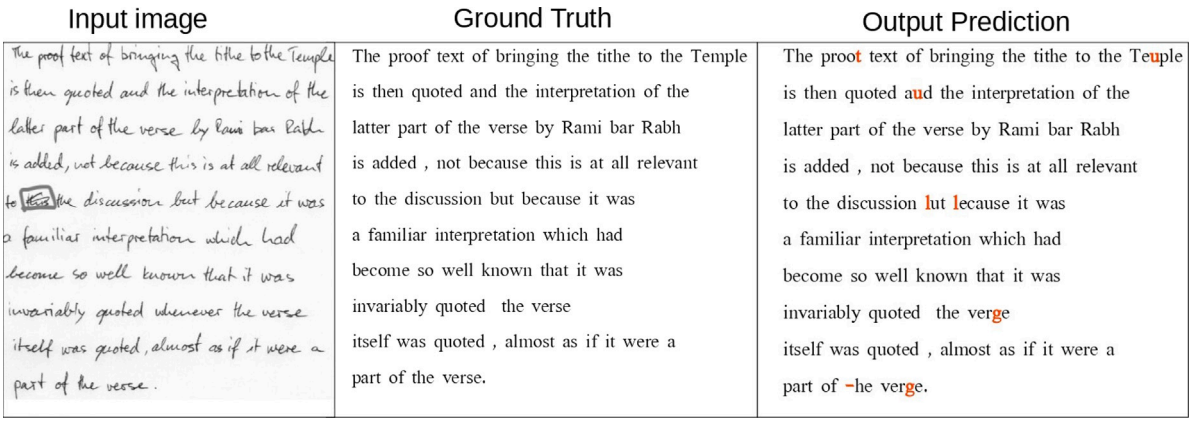


Fig. 5. Input image with the corresponding ground truth and its transcription from IAM dataset.

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] Coquenet 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. BLPnet: 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. Tablext: 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 Recogn (IJDAR) 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] Coquenet 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 multidimensional 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] Coquenet 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. Convolve, 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.