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].

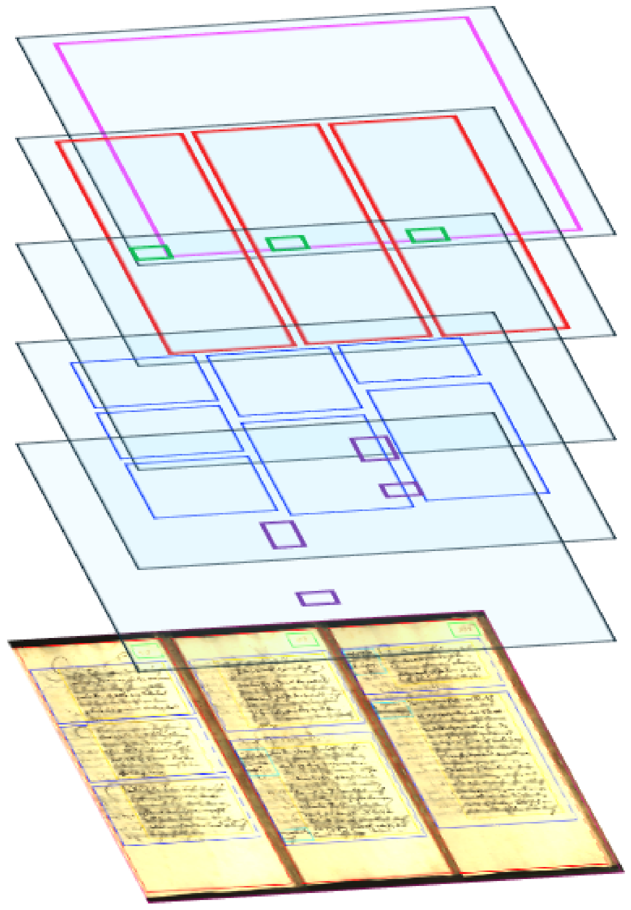
2) Segmentation-free approaches: To alleviate the need for line-level segmentation, some approaches have been developed to handle single-column pages or paragraphs. Yousef *et al.* [25] and Coquenet *et al.* [30] transformed the 2D image problem into a one-dimensional problem and used the Connectionist 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)

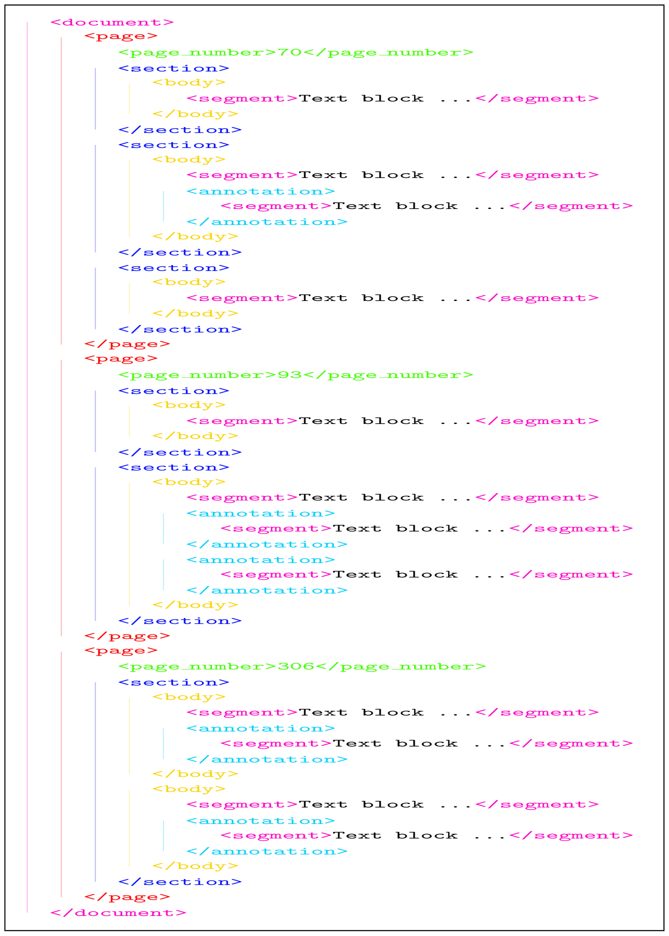
DLA aims to identify and categorize regions of interest in a document image. The field has evolved significantly, from traditional rule-based methods to modern deep learning approaches [32]. Current approaches can be broadly categorized into two main paradigms: pixel-based classification approaches and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

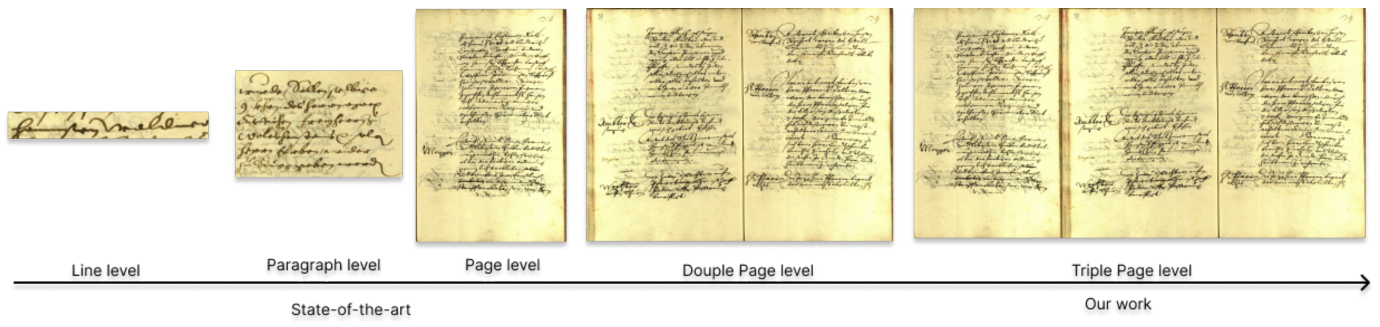


Fig. 2: Document recognition complexity across multiple scales: from line-level to triple-page documents. (a) Line level, (b) Paragraph level, (c) Single-page document, (d) Double-page document, (e) Triple-page document. Images are from the READ 2016 dataset.

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:

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

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:

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

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:

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

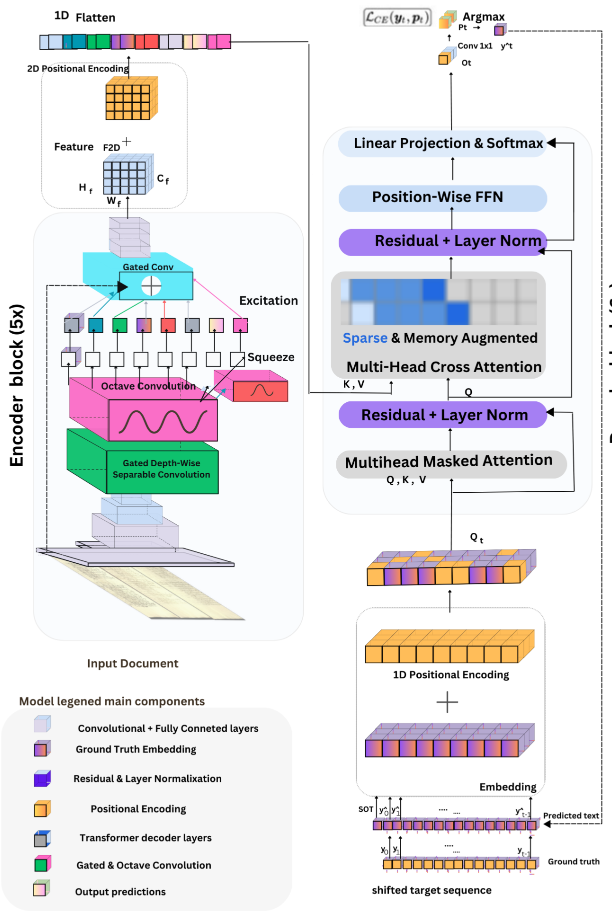


Fig. 3: Overview of the HAND Architecture: The HAND integrates convolutional layers as encoder for spatial feature extraction and a transformer decoder layers as a decoder for sequential prediction.

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 $\boldsymbol{f}_4$.

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

The next step involves Gated Convolution combined with an FCN:

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

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.

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

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

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

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.

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

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.

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

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.

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

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.

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

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.

$$\mathbf{C} = \sum_{l} \lambda_l \cdot \mathrm{MultiHeadAttn}_l(Q, K_l, V_l) \tag{14}$$

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.

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

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.

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

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: $\mathcal{M}_{\text{line}}$, $L = 5$, $E_l$, $\alpha_0$, $\rho_0$, $G$, $B_0$
1: $\theta_e \leftarrow$ InitializeWeights($\mathcal{M}_{\text{line}}$) {Initial encoder}
2: $\theta_d \leftarrow$ RandomInitialize() {Initial decoder}
3: $\phi \leftarrow$ InitializeComplexityNetwork() {Initial MSAP}
4: $W_0 \leftarrow \theta_e, \theta_d, \phi$ {Initial weights}
5: for $l \in L$ do
6:     $W_l \leftarrow W_{l-1} + \Delta W_l$ {Transfer learning (Eq. 20)}
7:     $\theta, \phi \leftarrow$ AdaptParameters($l, W_l$) {Current level}
8:     $B_l \leftarrow \max(\lfloor B_0 \cdot \gamma^l \rfloor, B_{min})$ {Adapt batch(Eq. 21)}
9:     $X_l^{syn} \leftarrow G_l(z; \theta_l)$ {Synthetic data (Eq. 22)}
10:    $\mathcal{D}_l \leftarrow$ PrepareDataset($l, X_l^{syn}$) {Comb real synth}
11:    for $e = 1$ to $E_l$ do
12:        $C_l \leftarrow$ AssessComplexity($\mathcal{D}_l, \phi$)
13:        $\alpha_l \leftarrow$ ComputeCurriculumWeight($l, e$) {Eq. 19}
14:        for $b \in \mathcal{D}_l$ do
15:            $x'_i \leftarrow$ ApplyAugmentation($b$)
16:            $F_1 \leftarrow$ FirstPassFeatures($x'_i, \theta, e$) {Algo 2}
17:            $F_2 \leftarrow$ SecondPassFeatures($F_1, C_l$) {Algo 3}
18:            $\mathcal{L}_l \leftarrow$ ComputeLosses($F_1, F_2, b$)
19:            $\mathcal{L}_{\text{total}} \leftarrow \alpha_l \mathcal{L}_l$ {Apply curriculum weight}
20:            $\theta, \phi \leftarrow$ UpdateParameters($\mathcal{L}_{\text{total}}, \theta, \phi$)
21:        end for
22:        $\alpha, \rho \leftarrow$ AdjustHyperparameters($e$)
23:    end for
24:    $W_l^* \leftarrow \{\theta, \phi\}$ {Save best model for current level}
25: end for
26: return $W_{\text{final}}$ {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 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: $X$, $\theta$, $e$
    Initialize: $E_{\text{warmup}} = 150$, $\alpha_0 = 0.1$, $\gamma = 0.5$
 1: $\boldsymbol{f}_{\text{base}} \leftarrow \text{FCN}(\text{Conv2D}(X))$ {Initial features Eq.1}
 2: $\boldsymbol{f}_{\text{gated}} \leftarrow \text{GatedDSConv}(f_{\text{base}})$ {Depth-wise Eq. 2}
 3: $\boldsymbol{f}^H_{\text{oct}}, f^L_{\text{oct}} \leftarrow \text{OctaveConv}(f_{\text{gated}})$ {Freq./decom.Eq. 3}

 4: $\boldsymbol{f}_{\text{se}} \leftarrow \text{SE}(f^H_{\text{oct}}, f^L_{\text{oct}})$ {Channel recalib. Eq. 4}
 5: $PE \leftarrow \text{PositionalEncoding2D}(f_{\text{se}})$ {Eq. 6}
 6: $\alpha_e \leftarrow \alpha_0(1 + \gamma \min(1, e/E_{\text{warmup}}))$ {Eq. 26}
 7: $\boldsymbol{f}_1 \leftarrow f_{\text{se}} + \alpha_e PE$ {Eq. 25}
 8: return $f_1$
```

Algorithm 3 Second Pass Feature Processing

```
Input: $\boldsymbol{f}_1$, $C_l$, $l$, $M$
    Initialize: $\lambda_{\text{mem}} = 0.5$, $\lambda_{\text{sparse}} = 0.5$
 1: $\alpha_{C_l}, \beta_{C_l} \leftarrow \text{ComputeScalingFactors}(C_l)$ {Eq. 28}
 2: $P^M_{\text{doc}} \leftarrow \text{DocumentContextEncoding}(f_1)$
 3: $R^i_M \leftarrow \text{RelativePositionEncoding}(f_1)$
 4: $q \leftarrow E(y^i_M) + \alpha_{C_l} P^M_{\text{doc}} + \beta_{C_l} R^i_M$ {Eq. 27}
 5: $\omega_{C_l} \leftarrow \text{ComputeComplexityWeight}(C_l)$ { For Eq. 29}
 6: $A_M \leftarrow \text{MemoryAugmentedAtten}(q, f_1, M)$ {Eq. 11}
 7: $A_S \leftarrow \text{SparseAttention}(q, f_1)$ {Eq. 12}
 8: $\boldsymbol{f}_2 \leftarrow \text{MultiHeadAtten}([A_M; A_S], f_1, \omega_{C_l})$ {Eq. 29}
 9: $\boldsymbol{f}_2 \leftarrow f_2 \cdot \alpha_{C_l}$
10: return $f_2$
```

where $E(y^i_M)$ represents token embeddings for content understanding, P^M_{doc} captures document-level context, and R^i_M 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:

$$\text{h}_i = \text{Attention}(QW^Q_i, KW^K_i, VW^V_i, C_l) \tag{31}$$

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:

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

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

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:

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

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	
	$\text{mAP}_{\text{CER}} \uparrow$	LOER $\downarrow$	$\text{mAP}_{\text{CER}} \uparrow$	LOER $\downarrow$	$\text{mAP}_{\text{CER}} \uparrow$	LOER $\downarrow$
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$_{\text{CER}}$ 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$_{\text{CER}}$, 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

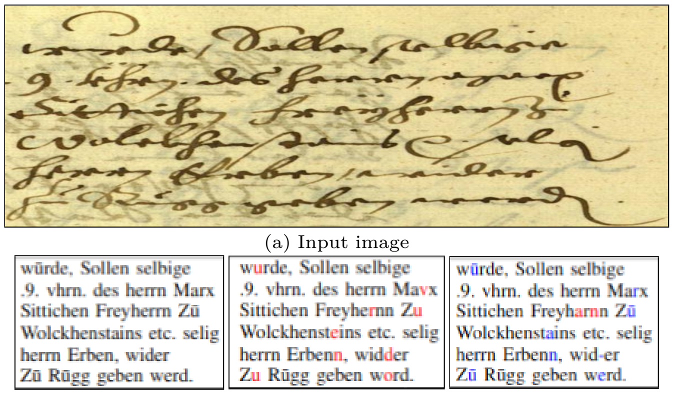
Method	Line		Paragraph	
	CER (%)	WER (%)	CER (%)	WER (%)
(CNN+BLSTM^a) [23]	4.66	-	-	-
(CNN+RNN) [61]	5.1	21.1	-	-
(CNN+MDLSTM) [61]	4.8	20.9	-	-
(ResneSt-Trans^b) [15]	-	-	4.20	8.75
(FCN+LSTM^c) [63]	4.10	16.29	-	-
DAN [9]	4.10	17.64	3.22	13.63
HAND	2.71	10.98	3.18	12.55
HAND+mT5	**1.65**	**4.95**	**2.13**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


(a) Input image

würde, Sollen selbige
.9. vhrn. des herrn Marx
Sittichen Freyherrn Zū
Wolckhenstains etc. selig
herrn Erben, wider
Zū Rūgg geben werd.

(b) Ground truth (GT)

wurde, Sollen selbige
.9. vhrn. des herrn Mavx
Sittichen Freyhernn Zu
Wolckhensteins etc. selig
herrn Erbenn, widder
Zu Rūgg geben word.

(c) HAND output

wūrde, Sollen selbige
.9. vhrn. des herrn Marx
Sittichen Freyharnn Zū
Wolckhenstains etc. selig
herrn Erbenn, wid-er
Zū Rūgg geben werd.

(d) HAND+mT5 output

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

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.

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,

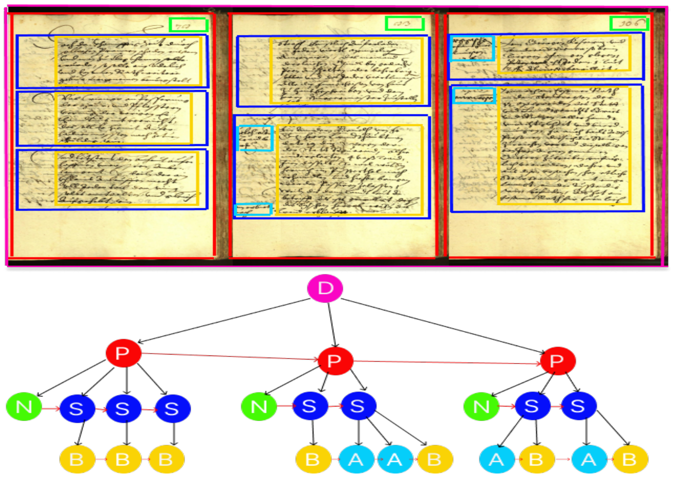


Fig. 5: Extended hierarchical structure of a triple-column document. Arrows indicate relationships among nodes: top-bottom indicates the hierarchical structure while left-right indicates the reading order.

significantly improving upon previous models such as DAN (5.17% single-page) and DANCER (3.37% single-page). The mean Average Precision (mAP$_{\text{CER}}$) 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$_{\text{CER}}$ 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:

$$q^i_M = E(y^i_M) + \alpha(C_l)P^M_{\text{doc}} + \beta(C_l)R^i_M \tag{S.6}$$

Equation S.14 represents a query mechanism that integrates three main components. Token embeddings $E(y^i_M)$ for content understanding, document-level context P^M_{doc} for structural awareness, and relative positional information R^i_M 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:

$$\text{h}_i = \text{Attention}(QW^Q_i, KW^K_i, VW^V_i, C_l) \tag{S.10}$$

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:

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

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:

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

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:

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

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:

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

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:

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

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

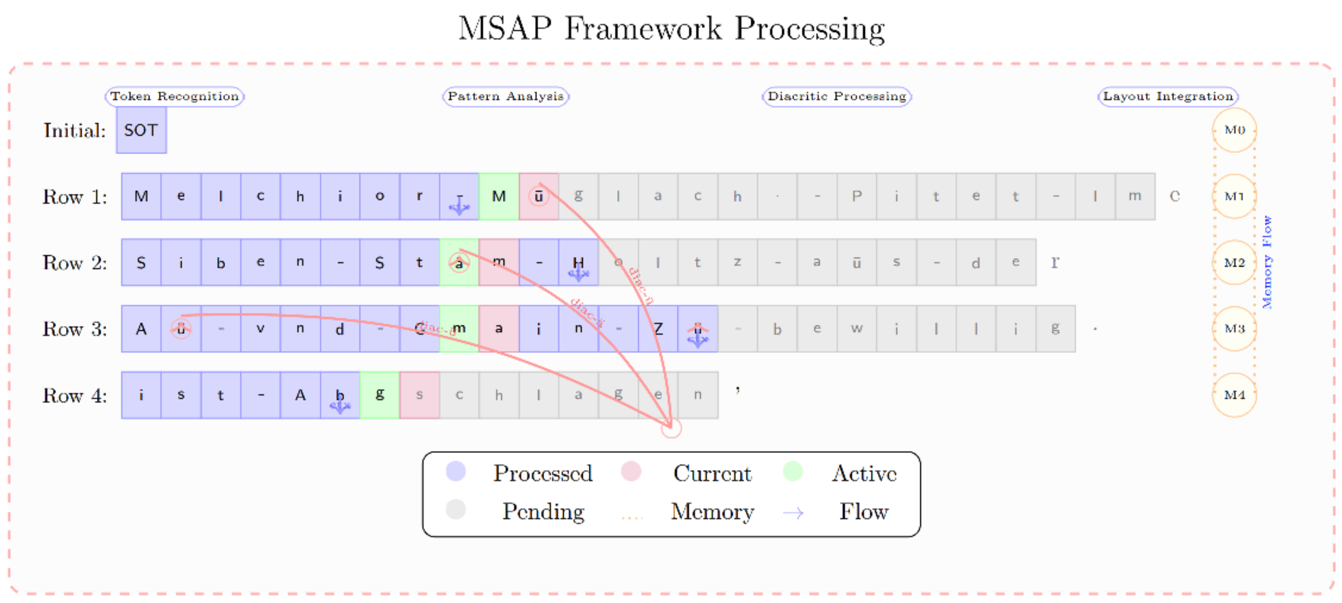


Fig. S.1: MSAP framework's progressive processing visualization. Highlighted features: (1) token-level operations with explicit memory flows, (2) diacritic pattern recognition (red connections between ū, ä), (3) hierarchical memory state transitions (M0-M4), (4) comprehensive layout integration. Color 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 $\phi(\mathring{u})$ 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 $\rightarrow$ Line $\rightarrow$ Layout
- Top-down: Layout $\rightarrow$ Line $\rightarrow$ Character

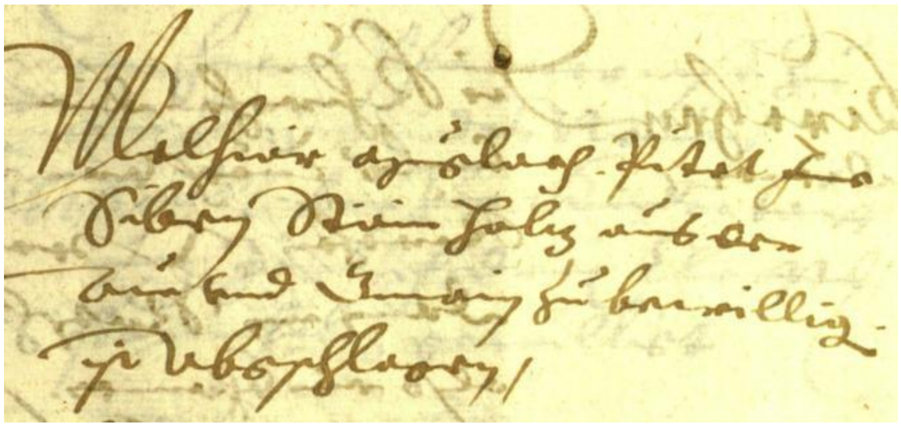


Fig. S.2: Representative historical German document showcasing complex characteristics that challenge context handling: (1) varied diacritical marks (ū, ä), (2) historical letter forms, (3) complex layout structure, and (4) varying character spacing.

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.

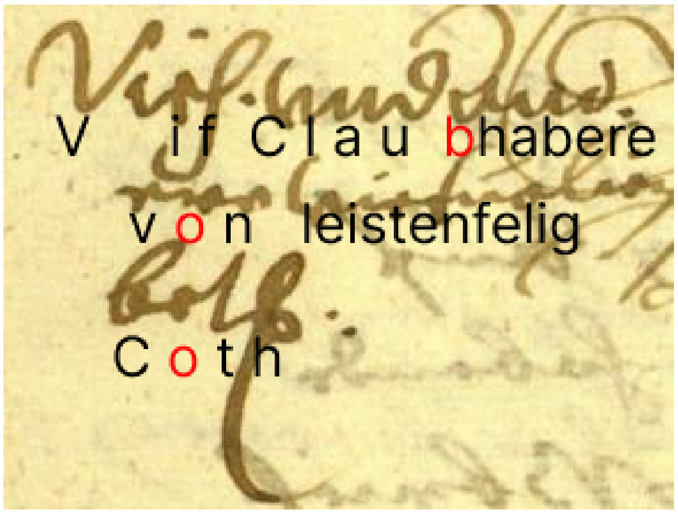

V if Clau bhabere
von leistenfelig
Coth

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 recognized 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: $R$, $M$, $\theta$, $D$ {HADN output, mT5 model, threshold, lexicon} Initialize: $Y \leftarrow \emptyset$, $C \leftarrow \emptyset$
 1: $O, L \leftarrow \text{ProcessHADN}(R)$
 2: for $l \in \{\text{sent}, \text{para}, \text{page}\}$ do
 3:     $S_l \leftarrow \text{Segment}(O, l, L)$
 4:     $ctx \leftarrow \text{Context}(S_l)$
 5:     for $s \in S_l$ do
 6:         $err \leftarrow \text{DetectErrors}(s, D)$
 7:         if $err \neq \emptyset$ then
 8:             $t \leftarrow \text{TokenizeSP}(s, ctx)$
 9:             $cand \leftarrow M(t, ctx)$
10:             $scr \leftarrow \text{Confidence}(cand)$
11:             for $i \leftarrow 1$ to $|cand|$ do
12:                 if $scr[i] > \theta$ then
13:                     $c' \leftarrow \text{Apply}(s, cand[i])$
14:                     if ValidLayout$(c', L)$ then
15:                         $s \leftarrow c'$
16:                         $C \leftarrow C \cup scr[i]$
17:                     end if
18:                 end if
19:             end for
20:         end if
21:         $Y \leftarrow Y \cup \text{Format}(s, L)$
22:     end for
23:     $Y \leftarrow \text{Reconcile}(Y, l)$
24: end for
25: $Y \leftarrow \text{Reconstruct}(Y, L, C)$
26: $Y \leftarrow \text{Validate}(Y, D)$
27: 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:

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

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

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: $\mathcal{O}(M d_f)$ for line and layout token detection
- Feature extraction: $\mathcal{O}(HW d_f)$ with adaptive selection based on complexity score:

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

Second Pass:

- Query generation: $\mathcal{O}(L d_{\text{model}})$ with adaptive scaling
- Complexity-aware attention: $\mathcal{O}(ML d_{\text{model}})$, 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 $\eta_{\text{cache}} = \frac{f_{\text{reuse}}}{f_{\text{total}}}$, 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.