NeuroInk: Retinomorphic Spiking Sequence Modeling for Handwritten Text Recognition

Xiubo Liang[1], **Jinxing Han**[1], **Hongyi Duan**[1], **Qizhen Jiang**[1], **Yuke Li**[2], **Haoqi Zhu**[2], **Yu Zhao**[1], **Hongzhi Wang**[1,2*]

[1] School of Software Technology, Zhejiang University, Ningbo, China
[2] NetEase Yidun AI Lab, Hangzhou, China

Abstract

Static-input Spiking Neural Networks (SNNs) commonly create temporal dynamics by repeating the same image over multiple timesteps. For handwritten text recognition, however, this convention is input-redundant: the image contains no native event stream, while recognition depends on faint stroke fragments, degraded ink boundaries, and monotonic left-to-right alignment. We introduce NeuroInk, a retinomorphic spike-oriented recognizer that converts static handwriting into a low-timestep sequence of temporally ordered ink evidence. Instead of replaying identical frames, NeuroInk performs temporal ink transduction with quantized gates and positive gate-delta current modulation, decomposing each text-line image into coarse stroke-mass support and fine stroke-boundary evidence. The resulting evidence sequence is injected as time-varying currents into an adaptive spiking visual hierarchy, where high-frequency enhancement and mid-level detail reinjection are used to preserve fine ink structures during spiking propagation. For Connectionist Temporal Classification (CTC) decoding, two-dimensional spiking maps are read out as width-wise columnar sequences and processed by CTC-aligned association circuits that combine local horizontal context, global association, and selective trace memory. On the LAM benchmark, NeuroInk achieves a character error rate (CER) of 2.4% on the validation split and 2.6% on the test split using only two timesteps and greedy CTC decoding, without an external language model or lexicon. These results support the view that static-input SNN sequence recognition can benefit from co-designing sensory temporalization, spiking visual abstraction, and monotonic sequence association.

1 Introduction

Handwritten Text Recognition (HTR) maps a text-line image to a character sequence, commonly through a visual encoder and Connectionist Temporal Classification (CTC) alignment [1]. Convolutional, recurrent, and Transformer-based recognizers have substantially improved segmentation-free text recognition [2–5]. Yet this dense image-to-sequence formulation does not explicitly expose a property that becomes central for spike-oriented computation: discriminative handwriting evidence is often sparse, local, and stroke-level. Weak strokes, broken contours, faded boundaries, small gaps, and stroke terminals can separate visually similar characters and perturb the monotonic alignments learned by CTC. Historical manuscripts amplify this difficulty, as paper texture, bleed-through, faded ink, stroke degradation, and collection-specific variation make such cues harder to extract reliably [6–9]. Thus, offline HTR can be viewed not only as dense visual recognition followed by decoding, but also as a problem of preserving degraded ink evidence and organizing it along the writing direction.

*Corresponding author: hongzhiwang@zju.edu.cn

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Spiking Neural Networks (SNNs) appear well matched to sparse handwriting evidence because active spikes can support sparse accumulation and temporal propagation [10, 11]. Recent Spiking Transformers have advanced spike-based visual recognition through spike-compatible attention, binary residual communication, efficient firing patterns, and hierarchical spiking representations [10, 12–14]. However, these models primarily target generic visual recognition rather than line-level transcription. In typical static-image SNNs, the same image is repeated across timesteps, so temporal variation comes from simulation dynamics rather than input evidence [15]. This repeated-input convention is poorly suited to offline HTR: the image has no native event stream, yet recognition benefits from temporally organized stroke evidence; it is sensitive to high-frequency ink structures, whereas spiking membrane dynamics may attenuate fine details [16]; and it requires CTC-aligned width-wise dependencies rather than image-level aggregation. Spike-HTR [17] introduced coarse-to-fine InkCoder evidence and a CTC-aware Length Reducer for handwriting.

A central design question in neuromorphic HTR is how sensory evidence enters the spiking architecture. We argue that a static text-line image should be converted into temporal evidence for spiking propagation that varies across timesteps, preserves stroke structure, and supports left-to-right character alignment. Repeating the same image over timesteps expands computation without separating complementary recognition cues. We therefore build on sensory temporalization: constructing temporally ordered evidence from a static input before spiking propagation. Consistent with static-to-temporal SNN encoding [18, 17], we extend this approach using quantized gates and positive gate-delta modulation while treating coarse stroke support and fine boundary structure as complementary evidence phases. Ordering these phases before the spiking pathway turns a static line image into a low-timestep evidence sequence rather than a redundant stack of frames, shifting the role of time in SNN recognition from simulation overhead to evidence organization.

We develop NeuroInk, a spike-oriented hybrid recognizer for historical HTR. As shown in Fig. 1, NeuroInk performs retinomorphic temporal ink transduction, decomposing a static line image into coarse stroke support and fine boundary evidence. The resulting temporal ink evidence is converted into time-varying currents that drive a visual hierarchy of adaptive gated leaky integrate-and-fire (GLIF) neurons [19], where sparse spiking states form the main visual representation. To retain fine ink structures during spiking propagation, NeuroInk includes stroke-preserving enhancement and detail reinjection, followed by CTC-aligned association circuits for character-level and longer-range context over width-wise sequences.

NeuroInk uses a hybrid spiking architecture. Its visual and sequence pathways rely on sparse spiking states, while sensory transduction and CTC readout use continuous operations. On the LAM validation split, NeuroInk achieves a character error rate (CER) of 2.4% using only two timesteps. Our evaluation compares temporal ink evidence with repeated-frame input, examines supervision and association components, and reports spike activity and operation-level estimates. Our contributions are summarized as follows:

- We extend sensory temporalization for static-input neuromorphic sequence recognition with quantized gates and positive gate-delta modulation.
- We instantiate this formulation for HTR with retinomorphic temporal ink transduction, decomposing static line images into coarse stroke support and fine boundary evidence for low-timestep propagation.
- We develop NeuroInk, a spike-oriented hybrid recognizer that couples temporal ink evidence with adaptive GLIF visual encoding, detail-oriented feature reinjection, columnar readout, and CTC-aligned association circuits.
- We evaluate recognition accuracy, repeated-frame controls, component ablations, spike activity, and operation-level estimates that distinguish binary-input-driven operators from continuous computation.

2 Related Work

2.1 Historical HTR and Degraded Stroke Evidence

Historical HTR differs from standard OCR and scene-text recognition because accurate transcription depends not only on sequence decoding, but also on preserving degraded stroke evidence before

alignment. CTC remains a standard objective for unsegmented line recognition [1], and convolutional, recurrent, and multidimensional recurrent recognizers provide important segmentation-free baselines [2, 3]. Autoregressive scene-text recognizers, pretrained Transformer-based OCR models, and recent HTR-specific architectures have further improved contextual decoding, visual-language modeling, and line-level representation learning [20, 4, 5, 21, 22].

Historical manuscripts introduce stronger degradation, script variability, and collection-specific artifacts. IAM, LAM, and READ2016 provide representative benchmarks [6–8], and recent assessments continue to highlight document artifacts and collection shifts as practical obstacles [23, 9]. Existing artificial neural network (ANN) pipelines can learn stroke-sensitive features, but they typically treat ink evidence as a dense visual signal rather than as a source of temporal input structure or sparse computation. This motivates a document-specific interface that converts degraded ink evidence into temporally ordered recognition input.

2.2 Spiking Visual Backbones

SNNs are attractive for sparse evidence streams because binary spike events can reduce dense multiply-accumulate operations to activity-dependent accumulations in spike-driven weighted operators, a long-standing motivation for energy-aware and neuromorphic computing [24–26]. Direct SNN training has been enabled by surrogate gradients and spatio-temporal backpropagation [27, 28], while residual architectures and adaptive neurons improve depth, optimization stability, and neuronal dynamics [29, 19, 30]. Recent spiking vision research has extended these foundations to Transformer-style visual backbones. Spikformer introduces spike-form query, key, and value tensors [31]; Spike-driven Transformer redesigns attention and residual pathways toward addition-dominated binary spike communication [10]; and QKFormer sparsifies token or channel mixing through spike-form Q-K attention [13].

At larger scale, Meta-SpikeFormer, SpikingResformer, SDT-V3, and Spikingformer improve multi-stage representation learning, firing efficiency, and scalable spiking backbone design [12, 32, 14, 33]. Complementary directions refine spike-driven conversion, addition-only attention, spatio-temporal aggregation, multi-scale fusion, and positional augmentation [34–40]. These works advance generic spiking vision, but line-level HTR imposes a different geometry: degraded stroke evidence has to remain usable through height-to-width sequence collapse and blank-heavy monotonic decoding.

2.3 Temporal Evidence for Spiking HTR

HTR requires reliable extraction of degraded ink before CTC decoding. Bipolar self-attention and spiking relative positional encoding indicate that binary spike attention often requires different treatment of polarity, normalization, and positional-distance constraints from ANN attention [41, 42]. Frequency-domain analysis further suggests that spiking neurons can act as network-level low-pass filters [16], motivating explicit preservation of weak strokes and fine contours for CTC recognition. Static-to-temporal encoding motivates task-dependent evidence schedules [18]. MI-TRQR instead quantifies and suppresses redundancy in spike features across steps [15]. In handwriting recognition, online/offline fusion similarly suggests that complementary stroke cues can be useful before high-level decoding [43]. Spike-HTR [17] introduced coarse-to-fine InkCoder evidence and a CTC-aware Length Reducer for blank-dominated widths. NeuroInk retains the post-encoder width positions and modulates retinal relay currents using quantized gates and their positive inter-step changes.

3 Method

Spike-oriented HTR must bridge a static-to-dynamic mismatch between degraded line images and multi-step spiking computation. Rather than obtaining temporal activity from repeated static presentation or premature hard thresholding, NeuroInk derives ordered ink-evidence gates that expose broad stroke support before finer boundary cues. These gates modulate shallow receptive-field features through a retinal-gated relay, after which hierarchical spike-state encoding, width-wise association circuits, and CTC decoding produce the transcription (Fig. 1).

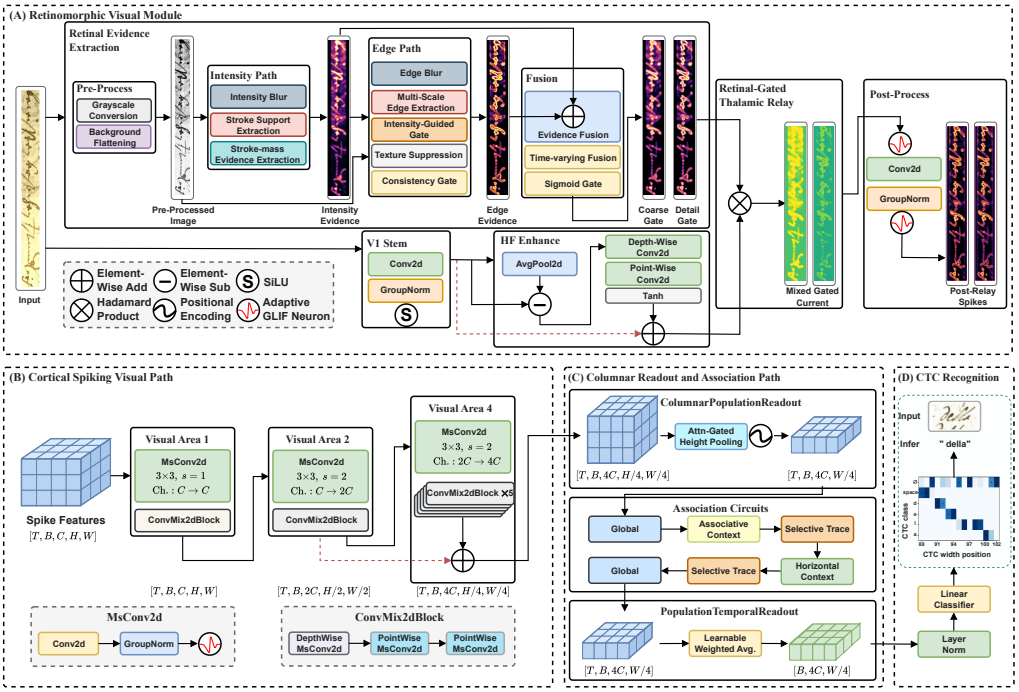


Figure 1: Overview of the NeuroInk architecture. (A) The retinomorphic visual module estimates time-varying retinal evidence from complementary intensity and boundary pathways, and converts this evidence into bounded gates that modulate receptive-field features through a retinal-gated relay. (B) The spike-oriented visual path processes the post-relay currents with multi-step synaptic convolutional blocks while postponing height collapse and preserving the writing-direction axis for sequence decoding. (C) The columnar readout collapses height into a width-indexed sequence, which is refined by global, associative, selective-trace, and horizontal context circuits before temporal population readout. (D) The CTC head maps width positions to character logits, from which the transcription is decoded.

3.1 Overview

Let $\mathbf{x} \in \mathbb{R}^{B\times C\times H\times W}$ denote a batch of line images, where B, C, H, and W are batch size, input channels, height, and width, respectively; the width axis follows the writing direction. NeuroInk first forms a shallow receptive-field representation, derives a temporally ordered retinal gate sequence, and combines them through relay currents:

$$\mathbf{F}_0 = \phi_{\text{stem}}(\mathbf{x}), \qquad \mathbf{G}_{1:T} = \mathcal{G}(\mathbf{x}), \qquad \mathbf{J}_{1:T} = \mathcal{R}(\mathbf{F}_0, \mathbf{G}_{1:T}). \tag{1}$$

Here, T is the number of simulation steps, ϕ_{stem} is a continuous receptive-field stem, $\mathcal{G}$ is the retinomorphic temporal transducer, and $\mathcal{R}$ is a retinal-gated relay that maps the static feature map and temporal gates to a sequence of driving currents. The stem and transducer remain continuous before spike-state conversion, which helps preserve weak ink evidence that may be lost under premature binarization. The resulting currents are processed by hierarchical spike-oriented visual encoding, height-collapsing columnar readout, width-wise association, temporal population readout, and CTC classification.

To make the spike-state computation explicit, we use $\mathrm{SN}(\cdot)$ to denote an adaptive spike-state transition and define a multi-step synaptic operator as

$$\tilde{\mathbf{X}} = \mathcal{N}_q(\mathcal{K}_q(\mathbf{X}))\,, \qquad \mathrm{Syn}_q(\mathbf{X}; \mathcal{K}_q) = \mathrm{SN}(\tilde{\mathbf{X}}) + \alpha_{\text{tr}}\tilde{\mathbf{X}}, \qquad q \in \{\text{1D}, \text{2D}\}. \tag{2}$$

Here, $\mathcal{K}_q$ is a 1D or 2D convolutional synapse, $\mathcal{N}_q$ is the corresponding normalization, and α_{tr} controls a learnable residual dense trace included in the low-timestep setting. This notation separates the spike-generating branch from the trace branch and clarifies which operators are organized around multi-step spike states. NeuroInk is therefore a spike-oriented hybrid OCR recognizer: dense evidence

construction is retained before spike-state conversion, while the post-relay visual pathway and the local or pointwise branches of the association blocks are formulated as multi-step synaptic operators. Compact dense or gated sequence mixers are kept for width-wise evidence propagation. Tensor resolutions, extractor details, surrogate gradients, and implementation constants are provided in Appendix A.

3.2 Retinomorphic Temporal Transduction

Historical handwriting often contains uneven stroke mass, low-contrast ink, bleed-through, paper texture, and broken contours. NeuroInk constructs spike-driving evidence from two complementary sources before spike-state conversion. The intensity branch estimates locally dark stroke support after compensating for slowly varying background, while the boundary branch emphasizes local edge responses that are important for thin contours, character-separating gaps, stroke endings, and diacritics. A local-support filtering step suppresses isolated texture responses while retaining boundary responses that co-occur with coherent low-contrast stroke evidence.

The two evidence maps are scheduled over a short spiking horizon and converted into bounded gates:

$$\begin{aligned} \mathbf{E}_t &= (1-\omega_t)\Psi_{\mathrm{int}}(\mathbf{x}) + \omega_t \Psi_{\mathrm{bd}}(\mathbf{x}), \\ \mathbf{G}_t &= Q_L[\sigma(\kappa_t(\mathbf{E}_t - \tau_t))]\,. \end{aligned} \tag{3}$$

Here, Ψ_{int} and Ψ_{bd} denote the intensity and boundary evidence extractors, ω_t controls their temporal mixture, τ_t and κ_t define the threshold and sharpness schedules, and Q_L maps the gate to L discrete levels with a straight-through estimator (STE) during training. Under this schedule, early gates are biased toward broad stroke support, whereas later gates become more boundary-selective; the spiking pathway therefore receives an ordered coarse-to-detail sequence rather than redundant static replays.

3.3 Spike-Oriented Visual-to-Sequence Encoding

The receptive-field pathway produces $\mathbf{F}_0$ with a lightweight high-frequency residual path that augments local stroke-detail responses before relay modulation. The retinal-gated relay then modulates the feature map with both persistent gate magnitude and newly emerging evidence:

$$\begin{aligned} \mathbf{J}_t &= \mathbf{F}_0 \odot (c_{\mathrm{tonic}} + c_{\mathrm{phasic}}\mathbf{A}_t)\,, \\ \mathbf{A}_t &= \mathrm{clip}_{[0,1]}\left(w_g \mathbf{G}_t + w_\Delta[\mathbf{G}_t - \mathbf{G}_{t-1}]_+\right), \qquad \mathbf{G}_0 = \mathbf{0}. \end{aligned} \tag{4}$$

The tonic term keeps a baseline receptive-field current, the gate-magnitude term follows stable ink evidence, and the positive-delta term responds to positive gate changes. With $\mathbf{G}_0 = \mathbf{0}$, the first-step difference equals $\mathbf{G}_1$; reversing gate order also changes relay currents. This design keeps a baseline current while allowing later boundary-biased gates to modulate the post-relay input.

Post-relay currents are converted into multi-step visual states by adaptive GLIF units. Here $\mathbf{I}_t$ denotes the layer-wise synaptic current, derived initially from $\mathbf{J}_t$. The membrane state and spike output are updated as

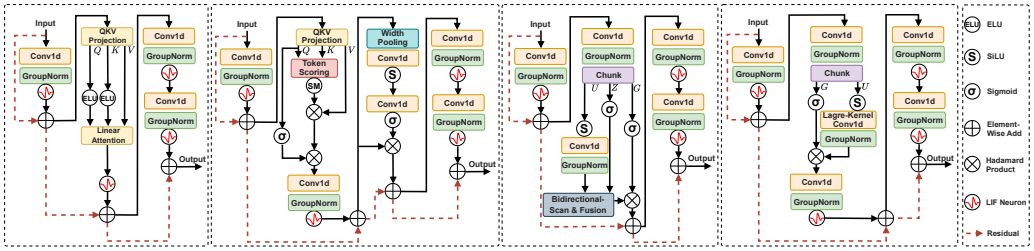
$$\mathbf{U}_t = \lambda \mathbf{U}_{t-1} \odot (1 - \rho\ \mathrm{stop}(\mathbf{S}_{t-1})) + \gamma \mathbf{I}_t, \qquad \mathbf{S}_t = \Theta(\mathbf{U}_t - \vartheta). \tag{5}$$

The decay λ, reset strength ρ, input gain γ, and threshold ϑ are learnable, and are channel-wise in the main configuration. The operator $\mathrm{stop}(\cdot)$ denotes stop-gradient in the reset branch. The forward pass uses the hard threshold $\Theta(\cdot)$, while gradients are propagated with a surrogate function specified in Appendix A. This update retains the integrate–fire structure while allowing different feature channels to use different temporal integration and reset dynamics.

The visual pathway contains three stages with increasing channel capacity and decreasing spatial resolution. Each stage uses multi-step synaptic convolutions for local stroke geometry and pointwise channel mixing, as illustrated in Fig. 1(B). A gated dual-resolution injection passes intermediate stroke-detail features to the final visual stage without replacing the higher-level spike-state representation. The final 2D visual state is collapsed along height by an attention-gated columnar readout, producing a width-indexed sequence aligned with the writing direction. A lightweight positional encoding is then added along the width axis before association.

3.4 Synaptic Association and CTC Training

After columnar readout, each simulation step t contains a sequence indexed by width position w. This sequence remains ambiguous because adjacent characters can share strokes, degraded gaps can



(a) Global View (b) Associative Context (c) Selective Trace (d) Horizontal View

Figure 2: Width-wise synaptic association modules in NeuroInk. After columnar readout, the visual state is represented as a width-indexed sequence along the writing direction. The four panels show the reusable sequence mixers used by the association pathway: (a) positive-kernel global integration, (b) associative context mixing, (c) selective bidirectional trace, and (d) large-kernel horizontal context. The main layout instantiates these mixers in the order (a) → (b) → (c) → (d) → (c) → (a).

collapse, and many latent CTC positions may correspond to blanks. NeuroInk therefore refines the sequence with width-wise association circuits that propagate evidence without making repeated dense width-by-width attention maps the primary sequence operator. Fig. 2 shows the association pathway in Fig. 1(C).

The association modules follow a common residual organization. For an input width sequence $\mathbf{H}$, a block applies a local synapse, a sequence mixer, and a pointwise channel integrator:

$$\begin{aligned} \mathbf{H}^{(1)} &= \mathbf{H} + \mathrm{Syn}_{\mathrm{1D}}(\mathbf{H}; \mathrm{DWConv}_{1d}^{k}), \qquad \mathbf{H}^{(2)} = \mathbf{H}^{(1)} + \mathcal{M}_{\mathrm{seq}}(\mathbf{H}^{(1)}), \\ \mathbf{H}^{+} &= \mathbf{H}^{(2)} + \mathrm{Syn}_{\mathrm{1D}}\Big(\mathrm{Syn}_{\mathrm{1D}}(\mathbf{H}^{(2)}; \mathrm{PWConv}_1); \mathrm{PWConv}_2\Big). \end{aligned} \tag{6}$$

Only the local and pointwise branches are written as $\mathrm{Syn}_{\mathrm{1D}}$; the middle branch is a compact width-wise mixer. The global mixer performs positive-kernel global integration over the width axis. The associative context mixer scores width positions, pools a compact line-level context, and redistributes it through token-dependent gates. The horizontal mixer uses large-kernel convolution to model local-to-mid-range continuity along the writing direction. The selective trace mixer performs gated bidirectional propagation and is defined below.

The selective trace modules are designed for CTC-style width sequences. They project the sequence into content, write-gate, and output-gate branches and perform bidirectional gated propagation:

$$\begin{aligned} \mathbf{r}_w^{\rightarrow} &= \delta \mathbf{r}_{w-1}^{\rightarrow} + \mathbf{z}_w \odot \mathbf{u}_w, \qquad \mathbf{r}_w^{\leftarrow} = \delta \mathbf{r}_{w+1}^{\leftarrow} + \mathbf{z}_w \odot \mathbf{u}_w, \\ \mathcal{M}_{\mathrm{trace}}(\mathbf{H})_w &= \mathbf{o}_w \odot \frac{\mathbf{r}_w^{\rightarrow} + \mathbf{r}_w^{\leftarrow}}{2}. \end{aligned} \tag{7}$$

Here, $\mathbf{u}_w$ is the content written at width position w, $\mathbf{z}_w$ is the write gate, and $\mathbf{o}_w$ is the output gate. The states $\mathbf{r}_w^{\rightarrow}$ and $\mathbf{r}_w^{\leftarrow}$ denote directional memory traces accumulated from the left and right contexts, respectively; the forward trace summarizes evidence from preceding width positions, while the backward trace summarizes evidence from following positions. The decay factor δ controls how far evidence is propagated along the writing direction. The learned write gate depends on sequence features and receives neither CTC blank posteriors nor explicit boundary masks. Together, the four mixers provide complementary width-axis transformations using linear, gated-scan, or convolutional operators rather than repeated dense self-attention over all width pairs.

After association, the multi-step sequence is fused by temporal population readout and classified at each width position:

$$\begin{aligned} \bar{\mathbf{H}} &= \sum_{t=1}^{T} \boldsymbol{\beta}_t \odot \mathbf{H}_t^{+}, \qquad \mathbf{O} = \boldsymbol{\xi} \odot \bar{\mathbf{H}} + (1 - \boldsymbol{\xi}) \odot \mathbf{H}_T^{+}, \\ \mathbf{L}_w &= \mathbf{W}_{\mathrm{cls}}\, \mathrm{LN}(\mathbf{O}_w) + \mathbf{b}_{\mathrm{cls}}. \end{aligned} \tag{8}$$

Here, $\boldsymbol{\beta}_t$ are weights normalized across simulation steps and $\boldsymbol{\xi}$ is a learnable channel gate that interpolates between the weighted temporal population and the final-step state. The model is optimized with CTC, which marginalizes over monotonic alignments between width positions and transcription labels. An auxiliary CTC branch is attached to an intermediate visual stage to provide

Table 1: Line-level HTR results on LAM. "External" denotes supervision or initialization beyond the LAM training split; * marks results reported in the HTR-VT benchmark.

Method	Regime	External	Decode	T	Params	CER (%)↓	WER (%)↓
CNN+BLSTM* [3]	ANN-CTC	No	CTC	–	9.3M	5.8	18.4
CRNN* [46]	ANN-CTC	No	CTC	–	18.2M	3.8	12.9
GFCN* [47]	ANN-CTC	No	CTC	–	1.4M	5.2	18.5
OrigamiNet-24* [48]	ANN-CTC	No	CTC	–	115.3M	3.0	11.0
ViT* [49, 5]	ANN-CTC	No	CTC	–	37.0M	6.1	19.1
HTR-VT [5]	ANN-CTC	No	CTC	–	53.5M	2.8	7.4
HTR-ConvText [50]	ANN-CTC	No	CTC	–	65.9M	2.7	**7.0**
TrOCR* [4]	Enc-Dec	Yes	AR	–	385.0M	3.6	11.6
Spike-HTR [17]	Spike-CTC	No	Greedy	2	76.7M	**2.5**	9.4
NeuroInk	Spike-CTC	No	Greedy	2	64.1M	2.6	9.1

direct supervision for mid-level stroke features [44, 45]. The final objective includes a symmetric Kullback–Leibler (KL) divergence, denoted $\mathcal{D}_{\mathrm{SKL}}$:

$$\mathcal{L} = \mathcal{L}_{\mathrm{CTC}}(\mathbf{L}, \mathbf{y}) + \lambda_{\mathrm{aux}} \mathcal{L}_{\mathrm{CTC}}(\mathbf{L}^{\mathrm{aux}}, \mathbf{y}) + \lambda_{\mathrm{kl}} \mathcal{D}_{\mathrm{SKL}}(\mathbf{L}, \mathbf{L}^{\mathrm{aux}}). \tag{9}$$

The auxiliary CTC term shortens the supervision path to the intermediate visual representation, and the symmetric posterior-consistency term aligns intermediate and final predictions without treating either branch as a fixed teacher. During inference, only the main CTC logits are decoded.

4 Experiments

We evaluate NeuroInk on line-level HTR using LAM, READ2016, and IAM. The main metrics are CER and word error rate (WER). Unless otherwise stated, NeuroInk results use target-only training, validation-based checkpoint selection, $T = 2$, and greedy CTC decoding without beam search, lexicon constraints, autoregressive decoding, or external language models. Full dataset handling, preprocessing, model instantiation, optimization settings, and loss constants are provided in Appendix B.1 and Appendix A.

4.1 Main Results

Table 1 compares NeuroInk with published line-level HTR systems on the LAM test set. For public baselines, we report the published test-set numbers. For NeuroInk, we select the checkpoint on the LAM validation split and decode greedily without an external language model or lexicon. Since public rows may differ in preprocessing, label normalization, checkpoint selection, and decoding details, we use CER as the primary metric for this restricted comparison: it measures character-level transcription fidelity and is less confounded than WER by word-boundary and tokenization differences under greedy, language-model-free decoding. Spike-HTR [17] reports a LAM test CER of 2.5%, while NeuroInk obtains 2.6% under a separate evaluation protocol. HTR-ConvText reports a lower WER (7.0% versus 9.1% for NeuroInk).

Table 2 summarizes NeuroInk validation and test performance on LAM, READ2016, and IAM under the same greedy-CTC evaluation protocol. The datasets differ in scale, language, and character inventory, providing a compact view of how the same model behaves across line-level HTR settings. NeuroInk reports 2.6% test CER on LAM. On READ2016 and IAM, NeuroInk attains test CERs of 4.2% and 5.6%, respectively. The controlled component analysis follows in Section 4.2. NeuroInk achieves competitive LAM test CER, while controls in Appendix B.3 (no-delta CER: 2.534%; full: 2.361%) and ablations in Section 4.2 probe how temporal evidence, supervision design, and width-wise association affect recognition.

4.2 Supervision and Association Ablations

We conduct controlled ablations on READ2016 to examine the supervision design of the intermediate CTC branch, and the capacity and context scale of the sequence-axis association pathway. Except for

Table 2: NeuroInk validation and test results under the unified greedy-CTC protocol. CER and WER are percentages; Charset excludes the CTC blank class.

Dataset	Split size			Language	Charset	Validation		Test	
	Train	Val	Test			CER↓	WER↓	CER↓	WER↓
LAM	19,830	2,470	3,523	Italian	89	2.4	8.7	2.6	9.1
READ2016	8,349	1,040	1,138	German	89	4.4	21.6	4.2	20.3
IAM	6,482	976	2,915	English	79	3.7	16.2	5.6	19.8

Table 3: READ2016 ablations: (a) intermediate supervision and temporal readout; (b) sequence association. CER, WER, and LAcc. (line accuracy) are percentages; Δ gives the CER change in percentage points from the corresponding Baseline. SKL denotes symmetric KL consistency.

(a) Supervision and readout

Variant	CER↓	Δ (pp) ↓	WER↓	LAcc.↑
Baseline	4.463	0.000	22.23	47.88
Weak aux.	4.458	-0.005	22.32	47.50
Aux. + SKL	**4.380**	**-0.083**	**21.56**	**49.81**
Channel-gated	4.492	+0.029	22.16	48.27
Dense trace	4.477	+0.014	22.01	48.27

(b) Association pathway

Variant	CER↓	Δ (pp) ↓	WER↓	LAcc.↑
Baseline	**4.463**	**0.000**	**22.23**	**47.88**
7-block dual-assoc.	4.531	+0.068	22.41	47.50
7-block late-global	4.536	+0.073	22.20	46.83
Large trace	4.681	+0.218	22.65	47.31
Horiz. $k = 47$	4.672	+0.209	23.26	45.87

the component being ablated, all variants use the same retinal ink-evidence transducer, spiking visual backbone, 64×512 input resolution, $T = 2$, hidden width 768, and greedy CTC decoding.

In Table 3 (a), the auxiliary CTC variant with symmetric posterior consistency gives the best result among the tested supervision settings. Aux. + SKL reduces CER from 4.463% to 4.380%, reduces WER from 22.23% to 21.56%, and raises line accuracy from 47.88% to 49.81%. In contrast, weakening the auxiliary signal changes CER only marginally, and the channel-gated or dense-trace temporal readouts do not improve CER in this READ2016 setting. We therefore use auxiliary CTC with symmetric KL consistency in the final configuration.

Table 3 (b) tests the effect of increasing association depth, trace capacity, or horizontal context width. The six-block Base setting remains best in CER: both seven-block layouts degrade recognition, while increasing trace capacity or the horizontal kernel size causes larger CER increases. These results suggest that, for the tested READ2016 configuration, balanced heterogeneous context is preferable to simply adding association capacity.

4.3 Qualitative and Error Analysis

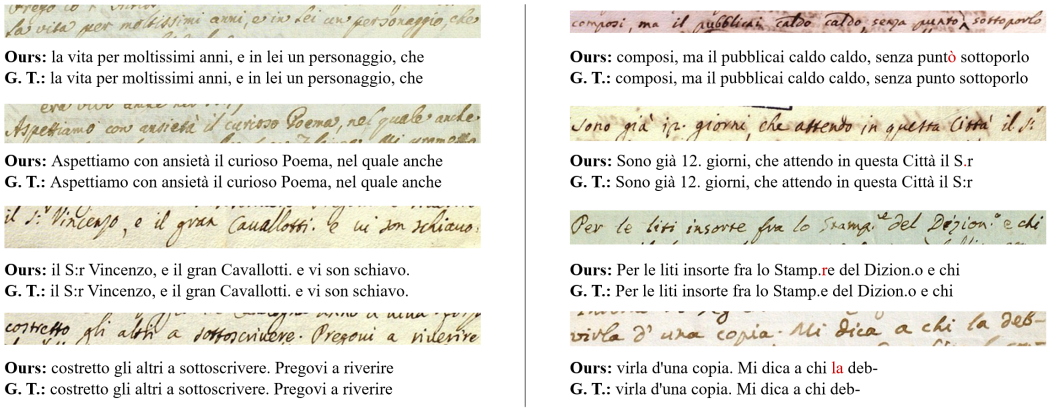
Fig. 3 highlights the main residual failure modes on LAM. The errors are not uniformly distributed across ordinary handwriting, but concentrate around transcription placeholders, diacritics, punctuation, abbreviation separators, and word-boundary decisions. This pattern is consistent with residual sensitivity to rare transcription markers, diacritics, and boundary-sensitive symbols, rather than broad failure on visually regular handwriting.

Table 4 quantifies this concentration: lines containing the placeholder "#" form only 5.99% of the validation set but contribute 31.55% of all character edits. In these cases, NeuroInk often replaces the placeholder with contextually plausible non-placeholder characters, which suggests a residual tension between contextual completion and literal preservation of transcription markers.

Table 4: Residual concentration around the placeholder "#".

Subset	Lines	CER	Edit share
Without "#"	2,322	1.72%	68.45%
With "#"	148	13.16%	31.55%

The remaining errors mainly involve local mark ambiguity and boundary-sensitive decisions. Local ambiguities include visually similar letter shapes and diacritic variants, especially between plain and accented vowels. Boundary errors appear around punctuation, abbreviation separators, apostrophes, and spaces, where a single insertion or deletion can produce a disproportionately large word-level penalty. Thus, the residual error profile is structured: lines without placeholder markers have



Ours: la vita per moltissimi anni, e in lei un personaggio, che
G. T.: la vita per moltissimi anni, e in lei un personaggio, che

Ours: composi, ma il pubblicai caldo caldo, senza puntò sottoporlo
G. T.: composi, ma il pubblicai caldo caldo, senza punto sottoporlo

Ours: Aspettiamo con ansietà il curioso Poema, nel quale anche
G. T.: Aspettiamo con ansietà il curioso Poema, nel quale anche

Ours: Sono già 12. giorni, che attendo in questa Città il S.r
G. T.: Sono già 12. giorni, che attendo in questa Città il S:r

Ours: il S:r Vincenzo, e il gran Cavallotti. e vi son schiavo.
G. T.: il S:r Vincenzo, e il gran Cavallotti. e vi son schiavo.

Ours: Per le liti insorte fra lo Stamp.re del Dizion.o e chi
G. T.: Per le liti insorte fra lo Stamp.e del Dizion.o e chi

Ours: costretto gli altri a sottoscrivere. Pregovi a riverire
G. T.: costretto gli altri a sottoscrivere. Pregovi a riverire

Ours: virla d'una copia. Mi dica a chi la deb-
G. T.: virla d'una copia. Mi dica a chi deb-

Figure 3: LAM qualitative examples. Red spans indicate character-level mismatches, with residual errors mainly appearing around placeholders, diacritics, punctuation, and word boundaries.

substantially lower CER, while the remaining failures are concentrated in historical-transcription edge cases requiring stricter preservation of placeholders, diacritics, punctuation, and word boundaries.

4.4 Energy and Spiking Activity

We estimate operation-level energy with the proxy commonly used in SNN analyses, where a multiply-and-accumulate operation is assigned 4.6 pJ and an accumulate operation is assigned 0.9 pJ [24]. This proxy gives a theoretical operation-count estimate; memory traffic, sparse scheduling overhead, and hardware utilization are not modeled. We follow the accounting convention used in recent spiking Transformer analyses: weighted computations before spike generation are counted as dense multiply-and-accumulate operations, whereas post-encoding weighted computations driven by spike activations are counted as firing-rate-weighted synaptic operations [31, 13, 33]. Because NeuroInk also contains continuous retinal modulation, relay gating, height readout, and sequence aggregation, these non-spiking functional cores are kept under dense operation accounting. Appendix C provides the full formulas and reports sensitivity under both idealized and stricter accounting protocols.

On 500 validation line images from LAM resized to 64×512, NeuroInk is estimated at 28.39 mJ per image, corresponding to a $29.52\times$ lower operation-level energy and a 96.61% reduction relative to the dense reference. Table 5 reports the block-wise firing rates measured on the same checkpoint and image subset. The second simulation step is consistently more active across both the visual encoder and the sequence mixer, consistent with the boundary-biased second gate producing denser post-relay activity. Appendix C contrasts this main estimate with a weight-only lower bound and a stricter hybrid estimate that assigns accumulate costs only to operators with recorded binary-spike inputs.

Table 5: Spike firing rates in NeuroInk blocks, averaged over measured spiking modules.

Block		Mean	t_0	t_1
Encoder	Stem	0.0857	0.0776	0.0939
	Stage 1	0.0867	0.0656	0.1078
	Stage 2	0.0740	0.0603	0.0877
	Stage 3	0.0679	0.0580	0.0779
Mixer	Association	0.1157	0.0993	0.1321
Average		0.0937	–	–

5 Conclusion

This work frames offline handwritten text recognition with spiking models as evidence temporalization rather than repeated simulation. NeuroInk instantiates this view by converting each static text-line image into a short ink-evidence sequence, where early activity emphasizes stroke support and later activity emphasizes boundary-sensitive details. The resulting temporal evidence drives a spike-oriented hybrid recognizer whose columnar representations are optimized with CTC for monotonic line transcription. Experiments show that a two-step retinomorphic evidence sequence can support competitive recognition on historical handwriting, while firing-rate measurements and operation-level estimates quantify the activity sparsity of the main spike-oriented pathway under the stated

accounting protocols. The LAM controls favor the full retinal-gating design over repeated static input and removal of positive gate-delta modulation. On LAM validation, placeholder-bearing lines contribute a disproportionate share of character edits, while selected examples show additional ambiguity in local marks and word boundaries. Overall, NeuroInk supports task-relevant evidence temporalization with positive gate-delta modulation for two-step spiking handwriting recognition.

6 Acknowledgement

This work was partly supported by Ningbo Key R&D Program (2025Z047) and Ningbo Major Application Demonstration Program (2025Z199).

References

[1] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006.

[2] Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(11):2298–2304, 2016.

[3] Joan Puigcerver. Are multidimensional recurrent layers really necessary for handwritten text recognition? In *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, volume 1, pages 67–72. IEEE, 2017.

[4] Minghao Li, Tengchao Lv, Jingye Chen, Lei Cui, Yijuan Lu, Dinei Florencio, Cha Zhang, Zhoujun Li, and Furu Wei. Trocr: Transformer-based optical character recognition with pre-trained models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 13094–13102, 2023.

[5] Yuting Li, Dexiong Chen, Tinglong Tang, and Xi Shen. Htr-vt: Handwritten text recognition with vision transformer. *Pattern Recognition*, 158:110967, 2025.

[6] U-V Marti and Horst Bunke. The iam-database: an english sentence database for offline handwriting recognition. *International journal on document analysis and recognition*, 5(1):39–46, 2002.

[7] Silvia Cascianelli, Vittorio Pippi, Martin Maarand, Marcella Cornia, Lorenzo Baraldi, Christopher Kermorvant, and Rita Cucchiara. The lam dataset: a novel benchmark for line-level handwritten text recognition. In *2022 26th International conference on pattern recognition (ICPR)*, pages 1506–1513. IEEE, 2022.

[8] Joan Andreu Sánchez, Verónica Romero, Alejandro H Toselli, and Enrique Vidal. Icfhr2016 competition on handwritten text recognition on the read dataset. In *2016 15th International conference on frontiers in handwriting recognition (ICFHR)*, pages 630–635. IEEE, 2016.

[9] Erez Meoded. Handwritten text recognition of historical manuscripts using transformer-based models. *arXiv preprint arXiv:2508.11499*, 2025.

[10] Man Yao, Jiakui Hu, Zhaokun Zhou, Li Yuan, Yonghong Tian, Bo Xu, and Guoqi Li. Spike-driven transformer. *Advances in neural information processing systems*, 36:64043–64058, 2023.

[11] Sicheng Shen, Dongcheng Zhao, Linghao Feng, Zeyang Yue, Jindong Li, Tenglong Li, Guobin Shen, and Yi Zeng. Step: A unified spiking transformer evaluation platform for fair and reproducible benchmarking. *Advances in Neural Information Processing Systems*, 38, 2026.

[12] Man Yao, JiaKui Hu, Tianxiang Hu, Yifan Xu, Zhaokun Zhou, Yonghong Tian, Bo Xu, and Guoqi Li. Spike-driven transformer v2: Meta spiking neural network architecture inspiring the design of next-generation neuromorphic chips. *arXiv preprint arXiv:2404.03663*, 2024.

[13] Chenlin Zhou, Han Zhang, Zhaokun Zhou, Liutao Yu, Liwei Huang, Xiaopeng Fan, Li Yuan, Zhengyu Ma, Huihui Zhou, and Yonghong Tian. Qkformer: Hierarchical spiking transformer using qk attention. *Advances in Neural Information Processing Systems*, 37:13074–13098, 2024.

[14] Man Yao, Xuerui Qiu, Tianxiang Hu, Jiakui Hu, Yuhong Chou, Keyu Tian, Jianxing Liao, Luziwei Leng, Bo Xu, and Guoqi Li. Scaling spike-driven transformer with efficient spike firing approximation training. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(4):2973–2990, 2025.

[15] Dengfeng Xue, Wenjuan Li, Yifan Lu, Yufan Liu, Wei Liu, Man Yao, Li Yang, Guoqi Li, Bing Li, Stephen Maybank, et al. Mi-trqr: Mutual information-based temporal redundancy quantification and reduction for energy-efficient spiking neural networks. *Advances in Neural Information Processing Systems*, 38:152480–152508, 2026.

[16] Yuetong Fang, Deming Zhou, Ziqing Wang, Hongwei Ren, Zecui Zeng, Lusong Li, Shibo Zhou, and Renjing Xu. Spiking neural networks need high-frequency information. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

[17] Xiubo Liang, Jinxing Han, Yuke Li, Haoqi Zhu, Yu Zhao, and Hongzhi Wang. Spike-htr: Spiking neural transformer for handwritten text recognition. *arXiv preprint arXiv:2608.01646*, 2026.

[18] Xue-Rui Qiu, Zhao-Rui Wang, Zheng Luan, Rui-Jie Zhu, Xiao Wu, Ma-Lu Zhang, and Liang-Jian Deng. Vtsnn: A virtual temporal spiking neural network. *Frontiers in neuroscience*, 17:1091097, 2023.

[19] Xingting Yao, Fanrong Li, Zitao Mo, and Jian Cheng. Glif: A unified gated leaky integrate-and-fire neuron for spiking neural networks. *Advances in Neural Information Processing Systems*, 35:32160–32171, 2022.

[20] Darwin Bautista and Rowel Atienza. Scene text recognition with permuted autoregressive sequence models. In *European conference on computer vision*, pages 178–196. Springer, 2022.

[21] Qianfeng Zhang, Feng Liu, and Wanru Song. Imtlm-net: improved multi-task transformer based on localization mechanism network for handwritten english text recognition. *Complex & Intelligent Systems*, 11(1):125, 2025.

[22] Changhun Kim, Martin Mayr, Thomas Gorges, Fei Wu, Mathias Seuret, Andreas Maier, and Vincent Christlein. Drethtr: Linear-time decoder-only retentive network for handwritten text recognition. *arXiv preprint arXiv:2602.17387*, 2026.

[23] Christel A Romein, Achim Rabus, Gundram Leifert, and Phillip Benjamin Ströbel. Assessing advanced handwritten text recognition engines for digitizing historical documents: Romein et al. *International journal of digital humanities*, 7(1):115–134, 2025.

[24] Mark Horowitz. 1.1 computing's energy problem (and what we can do about it). In *2014 IEEE international solid-state circuits conference digest of technical papers (ISSCC)*, pages 10–14. IEEE, 2014.

[25] Gwenevere Frank, Gopabandhu Hota, Keli Wang, Christopher Deng, Krish Arora, Diana Vins, Abhinav Uppal, Omowuyi Olajide, Kenneth Yoshimoto, Qingbo Wang, et al. Hiaer-spike software-hardware reconfigurable platform for event-driven neuromorphic computing at scale. *arXiv preprint arXiv:2602.18072*, 2026.

[26] Mingjing Li, Huihui Zhou, Xiaofeng Xu, Zhiwei Zhong, Puli Quan, Xueke Zhu, Yanyu Lin, Wenjie Lin, Xiaosha Li, Dong Wang, et al. A highly energy-efficient multi-core neuromorphic architecture for training deep spiking neural networks. *Nature Communications*, 2026.

[27] Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, and Luping Shi. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in neuroscience*, 12:331, 2018.

[28] Emre O Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019.

[29] Wei Fang, Zhaofei Yu, Yanqi Chen, Tiejun Huang, Timothée Masquelier, and Yonghong Tian. Deep residual learning in spiking neural networks. *Advances in neural information processing systems*, 34:21056–21069, 2021.

[30] Zijie Xu, Xinyu Shi, Yiting Dong, Zihan Huang, and Zhaofei Yu. Care-bn: Precise moving statistics for stabilizing spiking neural networks in reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pages 50749–50772, 2026.

[31] Zhaokun Zhou, Yuesheng Zhu, Chao He, Yaowei Wang, Shuicheng Yan, Yonghong Tian, and Li Yuan. Spikformer: When spiking neural network meets transformer. *arXiv preprint arXiv:2209.15425*, 2022.

[32] Xinyu Shi, Zecheng Hao, and Zhaofei Yu. Spikingresformer: Bridging resnet and vision transformer in spiking neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5610–5619, 2024.

[33] Chenlin Zhou, Liutao Yu, Zhaokun Zhou, Han Zhang, Jiaqi Wang, Huihui Zhou, Zhengyu Ma, and Yonghong Tian. Spikingformer: A key foundation model for spiking neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 2236–2244, 2026.

[34] Sangwoo Hwang, Seunghyun Lee, Dahoon Park, Donghun Lee, and Jaeha Kung. Spike-dattention: Training-free and fully spike-driven transformer-to-snn conversion with winner-

oriented spike shift for softmax operation. *Advances in Neural Information Processing Systems*, 37:67422–67445, 2024.

[35] Yufei Guo, Xiaode Liu, Yuanpei Chen, Weihang Peng, Yuhan Zhang, and Zhe Ma. Spiking transformer: Introducing accurate addition-only spiking self-attention for transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 24398–24408, 2025.

[36] Donghyun Lee, Yuhang Li, Youngeun Kim, Shiting Xiao, and Priyadarshini Panda. Spiking transformer with spatial-temporal attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13948–13958, 2025.

[37] Tianqing Zhang, Kairong Yu, Xian Zhong, Hongwei Wang, Qi Xu, and Qiang Zhang. Staa-snn: Spatial-temporal attention aggregator for spiking neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13959–13969, 2025.

[38] Wei Hua, Chenlin Zhou, Jibin Wu, Yansong Chua, and Yangyang Shu. Msvit: Improving spiking vision transformer using multi-scale attention fusion. *arXiv preprint arXiv:2505.14719*, 2025.

[39] Tao Deng, Chengfan Yang, Wenbo Liu, Chunyu Zhao, Ran Li, and Yi Huang. Transpikformer: Enhanced transformer-based spiking neural networks with uniform attention augmentation and depthwise convolutional positional encoding. *Knowledge-Based Systems*, page 115258, 2026.

[40] Naichuan Zheng, Yuchen Du, Hailun Xia, and Zeyu Liang. Signal-sgn: A spiking graph convolutional network for skeleton action recognition via learning temporal-frequency dynamics. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pages 1539–1548, 2025.

[41] Shuai Wang, Malu Zhang, Jingya Wang, Dehao Zhang, Yimeng Shan, Jieyuan Zhang, Yichen Xiao, Honglin Cao, Haonan Zhang, Zeyu Ma, et al. Bipolar self-attention for spiking transformers. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

[42] Changze Lv, Yansen Wang, Dongqi Han, Yifei Shen, Xiaoqing Zheng, Xuanjing Huang, and Dongsheng Li. Toward relative positional encoding in spiking transformers. *arXiv preprint arXiv:2501.16745*, 2025.

[43] Ayush Lodh, Ritabrata Chakraborty, Shivakumara Palaiahnakote, and Umapada Pal. A transformer based handwriting recognition system jointly using online and offline features. In *Asian Conference on Pattern Recognition*, pages 250–264. Springer, 2025.

[44] Chenxiang Ma, Rui Yan, Zhaofei Yu, and Qiang Yu. Deep spike learning with local classifiers. *IEEE Transactions on Cybernetics*, 53(5):3363–3375, 2022.

[45] Chenxiang Ma, Jibin Wu, Chenyang Si, Kay Chen Tan, et al. Scaling supervised local learning with augmented auxiliary networks. In *ICLR*, 2024.

[46] Théodore Bluche and Ronaldo Messina. Gated convolutional recurrent neural networks for multilingual handwriting recognition. In *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, volume 1, pages 646–651. IEEE, 2017.

[47] Denis Coquenet, Clément Chatelain, and Thierry Paquet. Recurrence-free unconstrained handwritten text recognition using gated fully convolutional network. In *2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 19–24. IEEE, 2020.

[48] Mohamed Yousef and Tom E Bishop. Origaminet: weakly-supervised, segmentation-free, one-step, full page text recognition by learning to unfold. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14710–14719, 2020.

[49] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[50] Pham Thach Thanh Truc, Dang Hoai Nam, Huynh Tong Dang Khoa, and Vo Nguyen Le Duy. Htr-convtext: Leveraging convolution and textual information for handwritten text recognition. *arXiv preprint arXiv:2512.05021*, 2025.

A Additional Method Details

This appendix provides the architectural and optimization details omitted from the main text for compactness. We first summarize the reported NeuroInk configuration, then describe the tensor resolutions, spike-state accounting boundary, association mixers, auxiliary losses, and CTC lattice visualization.

A.1 Concrete Model Instantiation

Table 6 summarizes the reported NeuroInk configuration. The model operates on line images resized to 64×512 and uses a two-step spiking horizon. The retinomorphic transducer emits an ordered pair of ink-evidence gates: the first step uses lower-threshold stroke-support evidence, while the second step increases boundary selectivity for narrow gaps, thin contours, diacritics, and stroke endings. Gates are represented with an eight-level quantized code, and gradients through the quantizer are approximated by a straight-through estimator during training.

Table 6: Implementation configuration of the NeuroInk model used in the reported experiments.

Component	Setting
Input resolution	64×512
Spiking horizon	$T = 2$
Retinal gate levels	$L = 8$
Retinal gate mode	quantized gate with STE
Retinal schedule	low-threshold support gate followed by boundary-biased gate
Retinal threshold schedule	$\tau_t \in [0.12, 0.56]$ with exponent 0.90
Retinal sharpness schedule	base slope 5.2 with decay coefficient 0.18
Time-varying boundary fusion	sigmoid schedule with bias -0.32 and slope 2.4
Relay tonic scale	$c_{\mathrm{tonic}} = 0.62$
Relay phasic scale	$c_{\mathrm{phasic}} = 0.38$
Relay gate / delta weights	$w_g = 0.65$, $w_\Delta = 0.35$
Spiking neuron	adaptive GLIF with channel-wise parameters
Surrogate slope	4.0
Visual channels	192, 384, 768
Visual depths	1, 1, 5
High-frequency stem residual	enabled, initial scale 0.05
Dual-resolution fusion	enabled, initial scale 0.20
Columnar readout	attention-gated height pooling with max-pooling correction
Width positional encoding	depthwise convolutional positional encoding, kernel 7
CTC sequence length	$W' = 128$
Association depth / heads	6 modules / 12 heads
Association MLP ratio	2.4
Association layout	Global–Associative Context–Selective Trace–Horizontal–Selective Trace–Global
Selective trace setting	state ratio 1.25, synapse kernel 9
Horizontal context setting	kernel 31, expansion ratio 1.8
Main temporal readout	weighted-gated readout
Auxiliary temporal readout	mean temporal aggregation
Auxiliary losses	auxiliary CTC (0.20) and symmetric KL consistency loss (0.05)
Dropout / stochastic depth	0.10 / 0.20

The base retinal evidence extractor is deterministic and parameter-free. It converts the input to grayscale, rescales intensities to a normalized range, estimates a local bright envelope for background flattening, and constructs complementary intensity- and boundary-based ink evidence maps. Sobel edge magnitude and multi-scale edge estimates provide boundary evidence, while stroke support, edge consistency, density-based filtering, and halo suppression are used to reduce isolated paper texture and scanning artifacts. The temporal gate schedule uses fixed base thresholds and slopes, with lightweight scalar calibration applied to gate sharpness. The receptive-field stem remains continuous before the first spike-state conversion, avoiding premature binarization of weak ink evidence. After retinal relay, the visual and association blocks operate on multi-step feature states through spike-oriented synaptic operators. NeuroInk is therefore a spike-oriented hybrid OCR architecture: deterministic evidence extraction, compact aggregation, normalization, temporal readout, and final classification remain

Table 7: Tensor resolutions for a 64×512 input line. The batch dimension is omitted, and T is shown where applicable.

Stage	Tensor shape	Role
Input image	$3 \times 64 \times 512$	line image
Retinal gates	$T \times 1 \times 64 \times 512$	ink-evidence gates
Stem	$192 \times 64 \times 512$	low-level features
Relay currents	$T \times 192 \times 64 \times 512$	gate-modulated currents
Post-relay spiking	$T \times 192 \times 64 \times 512$	spike-state conversion
Stage 1	$T \times 192 \times 64 \times 512$	high-resolution state
Stage 2	$T \times 384 \times 32 \times 256$	intermediate state
Stage 3	$T \times 768 \times 16 \times 128$	final 2D state
Columnar readout	$T \times 768 \times 128$	width sequence
Positional encoding	$T \times 768 \times 128$	width-axis encoding
Association	$T \times 768 \times 128$	sequence propagation
Temporal readout	768×128	aggregation over T
CTC logits	$128 \times \lvert\mathcal{V}\rvert$	CTC class logits

dense, whereas weighted operators in the post-relay visual and width-wise recognition pathways are organized around multi-step spike-valued states.

A.2 Tensor Resolutions and Operator Boundary

Table 7 reports tensor resolutions for an input line resized to 64×512. Let $\mathcal{V}$ denote the dataset-specific target character inventory together with the CTC blank class. The width axis is downsampled by a factor of four while preserving writing order, resulting in a main-branch CTC sequence length of 128 in the reported fixed-resolution setting.

For conservative M3 operation-level accounting, we distinguish spike-mediated weighted operators from dense components. The retinal evidence extractor, receptive-field stem, relay scaling, normalization layers, residual dense traces, height aggregation, temporal readout, and final classifier are treated as dense MAC-equivalent computation. Activity-dependent AC/SOP accounting is applied only to weighted post-relay visual and width-wise synaptic branches whose recorded presynaptic activations are binary spike-valued multi-step states. When a hybrid synaptic operator contains both a spike branch and a residual dense trace, the spike branch is activity-weighted while the trace remains dense. This boundary reflects the hybrid nature of NeuroInk rather than treating it as a fully event-driven system.

Adaptive GLIF parameters are constrained to valid numerical domains. The membrane time constant is initialized with $\tau_{\mathrm{m}} = 2.0$, while threshold, reset strength, and input gain are initialized to 1.0. The surrogate slope is $\alpha_{\mathrm{sg}} = 4.0$, and membrane states are reset at the beginning of each forward sequence. The sigmoid surrogate used for backpropagation is

$$\frac{\partial \mathbf{S}_t}{\partial \mathbf{U}_t} \approx \alpha_{\mathrm{sg}}\, p_t(1-p_t), \qquad p_t = \sigma(\alpha_{\mathrm{sg}}(\mathbf{U}_t - \vartheta))\,. \tag{10}$$

The auxiliary branch uses the intermediate visual stage and applies an analogous columnar readout. It then aggregates over time and aligns the auxiliary width length to the main CTC length before computing posterior consistency.

A.3 Association Mixer Details and Loss Constants

The main text defines the common residual template for association modules and the selective trace mixer. Table 8 reports the six-module association layout used in the reported configuration.

The global mixer performs linear sequence integration using the positive feature map $\phi(\cdot) = \mathrm{ELU}(\cdot)+1$:

$$\mathcal{M}_{\mathrm{glob}}(\mathbf{H})_w = \frac{\phi(\mathbf{Q}_w)^\top \sum_{u=1}^{W'} \phi(\mathbf{K}_u)\mathbf{V}_u^\top}{\epsilon + \phi(\mathbf{Q}_w)^\top \sum_{u=1}^{W'} \phi(\mathbf{K}_u)}. \tag{11}$$

Table 8: Width-wise association layout in NeuroInk.

Index	Module	Function
1	Global	linear sequence integration
2	Associative Context	token-gated global aggregation
3	Selective Trace	bidirectional gated propagation
4	Horizontal	large-kernel horizontal context
5	Selective Trace	gated propagation across gaps
6	Global	sequence-wide integration before readout

This provides sequence-wide context without materializing a dense width-by-width attention matrix; head and channel indices are omitted for compactness.

The associative context mixer scores tokens, pools a global context vector, and redistributes it through position-dependent gates:

$$\alpha_w = \mathrm{Softmax}_w\left(\frac{\langle \mathbf{Q}_w, \mathbf{K}_w \rangle}{\sqrt{d}}\right), \qquad \mathbf{c} = \sum_{w=1}^{W'} \alpha_w \mathbf{V}_w, \tag{12}$$

$$\mathcal{M}_{\mathrm{assoc}}(\mathbf{H})_w = \boldsymbol{\gamma}_w \odot \mathbf{c}, \tag{13}$$

where $\boldsymbol{\gamma}_w$ is a position-dependent gate predicted from the input sequence. The horizontal mixer is implemented as a large-kernel depthwise operator along the width axis, followed by pointwise projection and gating. Its role is to model local-to-mid-range handwriting continuity along the width axis.

For posterior consistency, the main and auxiliary logits are aligned to the same sequence length and converted to temperature-scaled posterior distributions over the CTC class dimension. With temperature τ_{kl}, the symmetric KL term is

$$\mathbf{p} = \mathrm{Softmax}(\mathbf{L}/\tau_{\mathrm{kl}}), \qquad \mathbf{q} = \mathrm{Softmax}(\mathbf{L}^{\mathrm{aux}}/\tau_{\mathrm{kl}}), \tag{14}$$

$$\mathcal{D}_{\mathrm{SKL}}(\mathbf{L}, \mathbf{L}^{\mathrm{aux}}) = \frac{\tau_{\mathrm{kl}}^2}{2}\left[\mathrm{KL}(\mathbf{p} \,\|\, \mathbf{q}) + \mathrm{KL}(\mathbf{q} \,\|\, \mathbf{p})\right]. \tag{15}$$

In the reported instantiation, $\tau_{\mathrm{kl}} = 1.0$. The fixed-weight objective is

$$\mathcal{L} = \mathcal{L}_{\mathrm{CTC}}(\mathbf{L}, \mathbf{y}) + 0.20\,\mathcal{L}_{\mathrm{CTC}}(\mathbf{L}^{\mathrm{aux}}, \mathbf{y}) + 0.05\,\mathcal{D}_{\mathrm{SKL}}(\mathbf{L}, \mathbf{L}^{\mathrm{aux}}). \tag{16}$$

The auxiliary CTC term provides a shorter supervision path to mid-level visual features. Because the KL term is symmetric, neither branch is treated as a fixed teacher. During inference, only the main CTC logits are decoded with greedy CTC collapse and no external language model.

A.4 Full-Line CTC Probability Lattice

To further interpret the CTC readout, Fig. 4 visualizes the complete width-axis CTC probability lattice for a correctly recognized validation line from LAM. The CTC head produces logits over $|\mathcal{V}|$ classes at each of the 128 width positions. For readability, the figure displays the blank token and a subset of classes that occur in the target transcription, occur in the predicted transcription, or attain high probability at one or more width positions. The visualization is a posterior lattice from the CTC head, not an attention map or an autoregressive decoding trace.

The lattice illustrates the sparse monotonic alignment pattern expected from CTC decoding. High blank probabilities occupy most width positions because the blank symbol accounts for background regions and transitions between character emissions. Non-blank classes, including letters, word-space symbols, and the final hyphen, appear as localized peaks along the writing direction. The greedy CTC path π is converted to a line transcription by

$$\mathcal{B}(\pi) = \mathrm{RemoveBlank}(\mathrm{MergeRepeats}(\pi)). \tag{17}$$

The displayed recognition result is therefore obtained by monotonic CTC collapsing of width-step posterior distributions, without autoregressive decoding or external language-model correction.

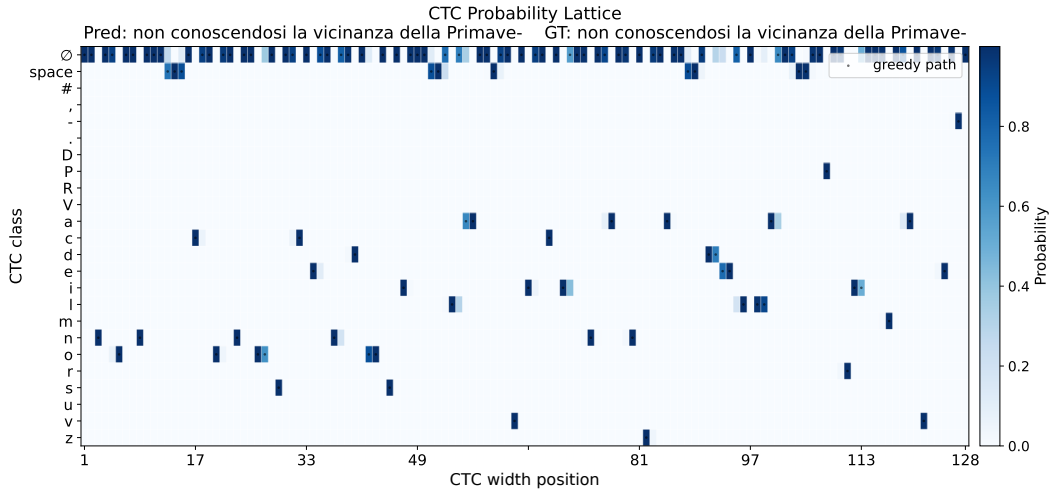


Figure 4: Full-line CTC probability lattice for a correctly recognized LAM validation line. The horizontal axis indexes CTC width positions, and the vertical axis lists the displayed CTC classes. Darker cells indicate higher predicted probabilities, and black dots denote the greedy CTC path. Most positions have high blank probability, whereas non-blank emissions appear as localized peaks. Applying CTC collapse to the greedy path yields “non conoscendosi la vicinanza della Primave-”, matching the corresponding ground-truth transcription, including the line-ending hyphen, in this example.

Table 9: Dataset split summary used in the reported NeuroInk experiments. Charset excludes the CTC blank class.

Dataset	Train	Val	Test	Language	Charset
LAM	19,830	2,470	3,523	Italian	89
READ2016	8,349	1,040	1,138	German	89
IAM	6,482	976	2,915	English	79

B Additional Experiments

This appendix provides the experimental protocol, dataset handling rules, optimization settings, reported ablation definitions, and mechanism visualizations that support the main experiments. Energy accounting is reported separately in Appendix C.

B.1 Experimental Protocol

READ2016, LAM, and IAM are evaluated at the line level. NeuroInk and all protocol-matched ablations use the same split, image preprocessing, character inventory construction, label normalization, and greedy CTC decoding rule within each dataset. When official training, validation, and test splits are available, we follow them. When a dataset requires a fixed custom split under our line-recognition protocol, the same split files are used for all protocol-matched NeuroInk variants.

Each line image is resized to height 64 while preserving aspect ratio and padded to a maximum width of 512. Samples exceeding the maximum resized width or violating the downsampled CTC length constraint are removed by a deterministic filtering rule before training and evaluation. The CTC input length is derived from the valid resized width after convolutional downsampling, and strict length checking is applied to avoid invalid target sequences. Predictions and references are normalized with the same Unicode normalization, character filtering, and whitespace-collapsing rules before metric computation.

Table 9 reports the final line counts used by the unified greedy-CTC protocol. The charset column counts target characters before adding the CTC blank symbol.

For LAM, the character inventory contains 89 target symbols, and the CTC output space contains 90 classes after adding the blank symbol. Text normalization uses NFKC normalization, whitespace collapsing, and removal of the marker "\not" from normalized labels. The same principle is applied to READ2016 and IAM with dataset-specific character inventories.

Character error rate and word error rate are computed from edit distance:

$$\mathrm{CER} = \frac{S_c + D_c + I_c}{N_c}, \qquad \mathrm{WER} = \frac{S_w + D_w + I_w}{N_w}, \tag{18}$$

where S, D, and I denote substitutions, deletions, and insertions, and N denotes the number of reference characters or words. WER is computed after the same normalization pipeline as CER, using whitespace-delimited word tokens.

Table 10 summarizes the default architectural and evaluation settings. The association layout abbreviates global, associative-context, selective-trace, horizontal-context, selective-trace, and global sequence circuits. The retinal front-end uses two simulation steps, an 8-level straight-through retinal gate, time-varying intensity/edge fusion, multi-scale edge evidence, edge-consistency filtering, and speckle suppression. The thalamic relay uses tonic and phasic current scales of 0.62 and 0.38, respectively, followed by post-relay spiking before the two-dimensional visual pathway.

Table 10: Default NeuroInk settings.

Item	Setting
Input resolution	64×512
Simulation steps	2
Hidden width	768
Visual depths	$(1, 1, 5)$
Visual channels	$(192, 384, 768)$
Association depth	6
Association layout	G/A/T/H/T/G
Parameters	64.05M
Objective	CTC + aux. CTC + KL
Decoder	Greedy CTC
Precision	mixed precision (fp16)

Training uses AdamW, cosine learning-rate decay, linear warmup, gradient clipping, automatic mixed precision (AMP), and an exponential moving average (EMA) of model weights. Auxiliary CTC supervision is attached to the mid-level visual stage, and a symmetric KL consistency term aligns the auxiliary and main CTC distributions. Table 11 reports the optimization settings used for the LAM configuration; dataset-specific runs use the same decoding and model-selection protocol unless otherwise specified.

Table 11: Training settings used for the reported LAM NeuroInk configuration.

Optimization and scheduling		Regularization and runtime	
Optimizer	AdamW	Precision	Native AMP, fp16
Learning rate	3.0×10^{-4}	TensorFloat-32	Enabled
Minimum learning rate	5.0×10^{-7}	Dropout	0.10
Warmup learning rate	8.0×10^{-6}	Stochastic depth	0.20
Weight decay	0.05	Auxiliary CTC weight	0.20
Schedule	Cosine decay	Consistency KL weight	0.05
Warmup	10 epochs	Consistency KL temperature	1.0
Epochs	125	Consistency KL type	Symmetric
Batch size	14	Seed	340
Validation batch size	24	Workers	8
Gradient clipping	1.0, norm mode	GPU	NVIDIA RTX 6000D
EMA	Enabled		

Training-time augmentation includes affine perturbation with probability 0.15, maximum rotation 2.5 degrees, translation ratio 0.015, and shear 2.5 degrees. Stroke perturbation, sharpening, additive noise, and width stretching are applied with probabilities 0.25, 0.10, 0.20, and 0.30, respectively. Width stretching samples a scale between 0.7 and 1.3. These augmentations are applied only during training; validation and test evaluation use deterministic resizing, padding, normalization, and greedy CTC decoding. Span masking is disabled in the reported configuration. Model selection is based on validation CER. EMA and non-EMA weights are evaluated under the same greedy CTC decoder, and the selected validation model is then used for the corresponding test-set evaluation.

Table 12: Controlled temporalization variants on the LAM validation split. All rows use the same backbone, $T = 2$, greedy CTC decoding, and no external language model or lexicon. ΔCER is reported in percentage points relative to Full NeuroInk.

Variant	CER (%)↓	ΔCER (pp)↓	WER (%)↓	LAcc. (%)↑
Repeated static frame, $T = 2$	2.611	+0.250	9.55	61.30
Same gate repeated twice	2.506	+0.145	9.13	62.39
Intensity-only gates	2.682	+0.321	9.89	60.53
Boundary-only gates	2.831	+0.470	10.54	59.07
Reversed temporal schedule	2.746	+0.385	10.18	59.80
No positive gate-delta relay	2.534	+0.173	9.27	62.02
Full NeuroInk	**2.361**	**0.000**	**8.66**	**63.36**

B.2 Reported Ablation Details

The controlled ablations in Table 3 use READ2016 and retain the same target data, image resolution, label normalization, character inventory, and greedy CTC decoding protocol. READ2016 is used for component analysis because it retains a degraded historical-document setting while making repeated controlled runs more tractable than LAM. Each ablation modifies only the component named in the table. The two panels use the same local reference row to keep the supervision and association comparisons interpretable; consequently, the association-pathway deltas isolate capacity and context-scale changes before adding the Aux. + SKL improvement selected for the final configuration.

The supervision ablations vary the intermediate CTC branch and temporal aggregation strategy. The weak-auxiliary variant reduces the influence of intermediate supervision, whereas Aux. + SKL uses the reported auxiliary CTC weight and symmetric posterior consistency. The channel-gated temporal readout variant replaces the reported weighted-gated population readout with channel-wise gating, and the dense-trace readout variant increases the dense temporal trace used during readout. These variants probe whether the observed recognition changes are associated with stronger mid-level supervision, temporal aggregation design, or dense temporal bypassing.

The association-pathway ablations vary sequence-axis capacity and context scale after columnar readout. The reported base layout uses six modules arranged as Global–Associative Context–Selective Trace–Horizontal–Selective Trace–Global. The seven-block dual-association and late-global variants add association capacity at different positions in the width-wise stack. The large-trace variant increases selective trace capacity, and the wide-horizontal variant enlarges the horizontal-context pathway. These variants probe whether CTC-axis recognition favors balanced heterogeneous context, larger memory traces, or broader horizontal smoothing under the local READ2016 reference setting.

For all ablations, validation and test metrics are computed with the same CER/WER implementation and greedy CTC collapse. No external language model, lexicon, or beam-search correction is used. Parameter counts are allowed to vary only as induced by the named architectural modification.

B.3 Temporalization Controls on LAM

Table 12 reports controlled variants designed to assess the contribution of retinal temporalization on the LAM validation split. All variants use the same NeuroInk backbone, $T = 2$, greedy CTC decoding, and no external language model or lexicon. The variants alter only the retinal evidence maps, their temporal order, or the positive gate-delta relay between timesteps. The repeated-frame control preserves the two-timestep computation while removing input-side retinal temporalization. The repeated-gate control retains spatial retinal gating but removes gate-level temporal variation. The intensity-only and boundary-only variants assess the complementarity of coarse stroke-support cues and boundary-sensitive cues. The reversed schedule probes sensitivity to evidence ordering, whereas the relay ablation removes the positive gate-delta term.

The controls indicate that the improvement of Full NeuroInk is not attributable to the two-timestep computation budget alone. Repeating the static image performs worse than Full NeuroInk, indicating that executing the same visual evidence for two timesteps is insufficient to match the full temporalized input. Repeating the same retinal gate also underperforms the full model, suggesting that gate-level

temporal variation provides useful information beyond spatial retinal gating alone. The single-source variants support the complementarity of intensity and boundary cues: using either source alone increases CER relative to Full NeuroInk, with the larger degradation observed for boundary-only gating. Reversing the temporal schedule further degrades recognition, suggesting sensitivity to the order in which coarse and boundary-sensitive evidence is presented; reversal also changes the gate-delta currents. Removing the positive gate-delta relay also reduces accuracy, suggesting that explicitly propagating positive gate changes contributes to the full temporalization design. With $\mathbf{G}_0 = \mathbf{0}$, removing the delta term also alters first-step current magnitude; this is a relay ablation, not a current-matched intervention.

B.4 Mechanism Visualizations and Scope

We visualize intermediate activations to inspect how the retinomorphic front-end and relay interface transform a static handwritten line into time-resolved spike evidence. The visualizations are computed from a trained checkpoint and displayed as line-image strips to preserve the handwriting aspect ratio. Unless otherwise stated, each map is normalized only for visualization and is not used as a recognition metric.

Fig. 5 visualizes the retinomorphic front-end on a held-out handwritten line. The first row shows the input line, the normalized grayscale image, and the complementary intensity and edge evidence maps. The intensity map emphasizes stroke-mass evidence, whereas the edge map emphasizes boundaries and contours. The second row shows fused ink evidence and the high-frequency-enhanced stem representation. The third row compares temporally allocated evidence and retinal gates at the two simulation steps. Under the default $T = 2$ configuration, the first step uses a lower evidence threshold and a more intensity-biased fusion, while the second step uses a higher threshold and stronger edge weighting. This implements a short-horizon coarse-to-detail evidence sequence rather than a repeated static input.

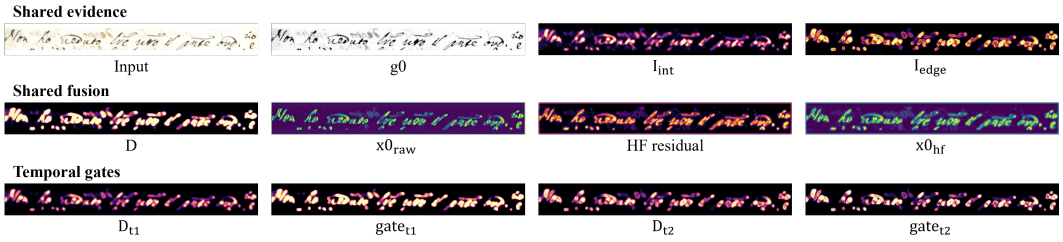


Figure 5: Retinomorphic front-end visualization. The first row shows the input handwritten line, normalized grayscale image, and complementary intensity and edge evidence maps. The second row shows fused ink evidence and the high-frequency-enhanced stem representation. The third row compares temporally allocated evidence and retinal gates at two timesteps. All maps are computed from intermediate activations of the trained model and preserve the original line-image aspect ratio.

Fig. 6 visualizes the gate-delta relay pathway. The first two rows are organized by timestep and follow the computation sequence from retinal gate to relay current, relay drive, and post-relay spike response. The final row compares the second-step response with a restricted relay variant that removes the positive gate-delta term while retaining gate-magnitude modulation. The visualization shows where late retinal-gate changes alter current modulation and post-relay spiking.

C Additional Energy Diagnostics

This appendix details the operation-level energy accounting used for NeuroInk and analyzes its sensitivity to different MAC/SOP conventions. The main text reports both a convention-aligned mixed MAC/SOP estimate and a stricter hybrid MAC/AC estimate. Following prior SNN energy analyses, we use the 45 nm CMOS operation costs $E_{\mathrm{MAC}} = 4.6$ pJ and $E_{\mathrm{AC}} = 0.9$ pJ [24]. These values define a theoretical operation-level proxy; they do not include memory movement, sparse scheduling overheads, control-flow overheads, nonlinear-function costs, or hardware utilization effects. All estimates are computed on 500 LAM validation line images resized to 64×512, using

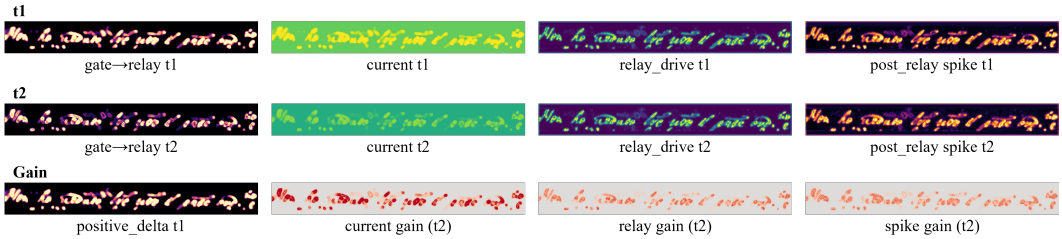


Figure 6: Gate-delta relay visualization. The first two rows show the relay pathway at two timesteps, following the order retinal gate, relay current, relay drive, and post-relay spike response. The last row shows second-step difference maps in current, relay drive, and post-relay spike response relative to a restricted relay variant without the positive gate-delta term.

the validation-selected LAM checkpoint. The block-level firing rates are reported in Table 5, while the estimates below use operator-level activity factors recorded during evaluation.

Let Ω denote the set of profiled weighted Conv/Linear operators, including depthwise, pointwise, and projection layers. Let MAC_l be the dense multiply-and-accumulate count of operator l on the time-expanded computation graph:

$$\mathrm{MAC}_l = T_l \mathrm{MAC}_l^{(1)}, \tag{19}$$

where $\mathrm{MAC}_l^{(1)}$ is the single-execution dense MAC count and T_l is the number of simulation steps or executions for operator l. Let r_l denote the mean presynaptic binary spike rate of operator l, averaged over images, executions, channels, and spatial or sequence positions in evaluation mode. For operators treated as spike-mediated, the synaptic operation count is estimated as

$$\mathrm{SOP}_l = r_l \mathrm{MAC}_l. \tag{20}$$

The dense all-MAC reference keeps the same operator graph and counts all profiled weighted and functional operations as dense MAC-equivalent operations:

$$E_{\mathrm{dense}} = E_{\mathrm{MAC}} \left(\sum_{l \in \Omega} \mathrm{MAC}_l + \mathrm{MAC}_{\mathrm{func}} \right). \tag{21}$$

Here, $\mathrm{MAC}_{\mathrm{func}}$ denotes profiled multiply-add-equivalent arithmetic outside Conv/Linear layers, such as continuous retinal modulation, relay gating, element-wise gates, and non-Conv/Linear aggregation operations. Relative energy gain and reduction are computed as

$$\mathrm{Gain} = \frac{E_{\mathrm{dense}}}{E_{\mathrm{method}}}, \qquad \mathrm{Reduction} = 1 - \frac{E_{\mathrm{method}}}{E_{\mathrm{dense}}}. \tag{22}$$

We report three accounting conventions and two functional-core-aware variants. The first convention is an idealized weight-only all-SOP protocol:

$$E_{\mathrm{M1}} = E_{\mathrm{AC}} \sum_{l \in \Omega} \mathrm{SOP}_l. \tag{23}$$

M1 is optimistic because it excludes dense functional cores and treats all profiled weighted operators as spike-mediated accumulations. It should therefore be interpreted as a lower-bound energy estimate and an upper-bound efficiency estimate. Adding dense functional cores yields

$$E_{\mathrm{M1b}} = E_{\mathrm{AC}} \sum_{l \in \Omega} \mathrm{SOP}_l + E_{\mathrm{MAC}} \mathrm{MAC}_{\mathrm{func}}. \tag{24}$$

M1b remains optimistic for hybrid operators whose weighted inputs are not recorded as binary spikes.

The second convention follows common SNN-Transformer accounting: input-encoding weighted operators are counted as dense MACs, while post-encoding weighted operators are counted as firing-rate-weighted SOPs for comparability with prior SNN energy reports:

$$E_{\mathrm{M2}} = E_{\mathrm{MAC}} \sum_{l \in \Omega_{\mathrm{input}}} \mathrm{MAC}_l + E_{\mathrm{AC}} \sum_{l \in \Omega \setminus \Omega_{\mathrm{input}}} \mathrm{SOP}_l, \tag{25}$$

Table 13: Theoretical operation-level energy estimates under different accounting protocols on 500 LAM validation line images.

Protocol	Est. energy (mJ/image)	Rel. gain	Reduction
Dense all-MAC reference	838.06	1.00×	0.00%
M1: weight-only all-SOP	21.24	39.45×	97.47%
M1b: all-SOP + functional MAC	21.94	38.19×	97.38%
M2: input-MAC + post-SOP	27.69	30.26×	96.70%
M2b: input-MAC + post-SOP + functional MAC	28.39	29.52×	96.61%
M3: conservative mixed MAC/AC	456.36	1.84×	45.55%

where Ω_{input} denotes the initial weighted image-encoding operators that operate before spike generation. The functional-core-aware variant is

$$E_{\text{M2b}} = E_{\text{MAC}} \sum_{l \in \Omega_{\text{input}}} \text{MAC}_l + E_{\text{AC}} \sum_{l \in \Omega \setminus \Omega_{\text{input}}} \text{SOP}_l + E_{\text{MAC}} \text{MAC}_{\text{func}}. \tag{26}$$

M2b is reported for convention-aligned comparison because it retains NeuroInk's explicitly profiled functional cores under dense-equivalent accounting while using SOP accounting for post-encoding spike-mediated weighted operators.

The third convention is a conservative mixed MAC/AC protocol. It partitions profiled weighted operators into binary-spike-input operators $\mathcal{S}$ and non-binary-input operators $\mathcal{A}$, with $\mathcal{S} \cup \mathcal{A} = \Omega$ and $\mathcal{S} \cap \mathcal{A} = \emptyset$. An operator is assigned to $\mathcal{S}$ only when its presynaptic activations are binary spike tensors in the recorded evaluation forward graph; otherwise, it is assigned to $\mathcal{A}$:

$$E_{\text{M3}} = E_{\text{AC}} \sum_{l \in \mathcal{S}} \text{SOP}_l + E_{\text{MAC}} \sum_{l \in \mathcal{A}} \text{MAC}_l + E_{\text{MAC}} \text{MAC}_{\text{func}}. \tag{27}$$

M3 is stricter than M2b because AC costs are assigned only when the recorded presynaptic activations are binary spikes; all non-binary weighted computation and continuous-valued functional cores remain under MAC-equivalent accounting.

Table 13 quantifies the sensitivity of the estimated operation-level energy to the MAC/SOP accounting convention. M1 is an idealized lower bound. M2 follows the convention-aligned SNN-Transformer estimate in which post-encoding weighted operators are treated as spike-triggered accumulations. M2b adds explicitly profiled functional cores to this convention-aligned estimate. M3 provides a stricter hybrid-model estimate that avoids assigning AC costs to non-binary weighted computation. Thus, M2b supports comparison with prior SNN energy reports, whereas M3 characterizes a more conservative deployment interpretation of the same model.

The gap between M2b and M3 quantifies the sensitivity of NeuroInk's estimated energy to whether hybrid post-encoding weighted operators are treated as SOPs or retained as dense MACs. As shown in Table 14, 46.32% of the profiled dense weighted-operation count is assigned to binary-spike-input operators under the conservative criterion, while 53.68% remains non-binary-input computation. This split is consistent with the architectural boundary used throughout the paper: NeuroInk is spike-oriented and activity-sparse in major post-relay branches, but it is not a fully event-driven recognizer.

Table 14: Conservative MAC/AC split of profiled weighted Conv/Linear operation counts under M3. Share denotes the fraction of dense weighted-operation counts before firing-rate scaling.

Operation type	Dense-op share	Accounting
Binary-spike-input	46.32%	AC-based SOP
Non-binary-input	53.68%	Dense MAC

Table 15 decomposes the estimated per-image operation energy by architectural stage. Under M2b, Stage3 + DualRes is the largest component, accounting for 47.93% of the estimated energy, followed by the initial stem with 21.53%. Under the stricter M3 protocol, Stage3 + DualRes accounts for 67.88% of the conservative mixed energy, followed by Stage2 with 18.41%. Under these proxy estimates, the dominant residual cost comes from the high-channel two-dimensional visual encoder, while the CTC readout and the one-dimensional association blocks contribute comparatively small fractions under both M2b and M3.

Table 15: Stage-wise estimated operation-level energy breakdown under dense all-MAC, M2b, and conservative M3 accounting. Values are reported per image.

Stage	All-MAC (mJ/img.)	M2b (mJ/img.)	M3 (mJ/img.)	M2b share	M3 share
Stem	6.11	6.11	6.11	21.53%	1.34%
V1/Stage1	145.77	3.43	25.68	12.07%	5.63%
V2/Stage2	116.95	2.90	84.00	10.21%	18.41%
V4/Stage3 + DualRes	528.59	13.61	309.77	47.93%	67.88%
Height readout	0.74	0.049	0.74	0.17%	0.16%
Positional encoding	0.007	0.001	0.007	$< 0.01\%$	$< 0.01\%$
Association blocks	39.39	2.22	29.55	7.83%	6.48%
Auxiliary head	0.45	0.072	0.45	0.25%	0.10%
Final readout	0.041	0.002	0.041	0.01%	0.01%

D Deployment Boundary and Hardware Path

The preceding energy estimates are operation-level proxies, not measurements from a deployed OCR accelerator. They should therefore be read as delimiting which parts of the forward graph are eligible for spike-aware accounting. NeuroInk is a spike-oriented hybrid recognizer: binary spike-state subgraphs are the components that may benefit from event-driven or sparse-accumulation execution, whereas retinal evidence construction, relay arithmetic, dense traces, normalization, compact readout, classification, and CTC collapse remain continuous or symbolic operations.

This distinction is reflected in the M2b and M3 protocols in Table 13. M2b uses a broader SNN-style accounting rule: post-encoding weighted operators are counted as firing-rate-weighted synaptic operations, while profiled functional cores remain dense MAC-equivalent operations. M3 uses a stricter operator-eligibility rule: AC/SOP costs are assigned only to operators whose recorded presynaptic activations are binary spikes, and non-binary weighted operators and continuous functional cores remain under MAC-equivalent accounting. The M2b–M3 gap therefore reflects the dependence of the proxy estimate on the chosen spike-operator boundary, rather than a measured hardware effect.

A possible implementation partitions fixed-point retinal preprocessing, spike-input GLIF and synaptic kernels, and dense sequence readout across suitable processing units. As in M3 accounting, AC/SOP eligibility requires recorded binary presynaptic inputs.

This partition remains a proposal, not a deployed system. Hardware measurements of latency and energy would require operator lowering, fixed-point calibration, and accounting for memory traffic, sparse scheduling, and host–accelerator transfers.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction describe the proposed spike-oriented handwritten text recognition architecture, the evaluated datasets and metrics, and the intended scope of the empirical claims. The recognition and efficiency claims are supported by the experimental results and ablations, and the energy claim is stated under the reported operation-level MAC/AC proxy rather than as measured hardware energy.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses the main limitations of the work, including the restricted evaluation scope of line-level handwritten text recognition, the use of an operation-level energy proxy instead of direct neuromorphic hardware measurements, and the hybrid rather than fully spike-driven nature of the architecture.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not present theoretical theorems or formal proof-based claims. The mathematical notation is used to describe the model architecture and training objective rather than to establish theoretical results.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides the architecture specification, dataset setup, preprocessing protocol, training configuration, decoding strategy, evaluation metrics, and ablation settings needed to reproduce the main experimental conclusions. Additional implementation and energy-estimation details are provided in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The evaluated datasets are existing public or externally available benchmarks, and the supplementary material includes anonymized code and instructions for reproducing the main training and evaluation pipelines. Dataset access and preprocessing steps are described in the experimental setup.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The experimental section specifies the datasets, input resolution, metrics, decoding protocol, model configuration, optimizer and training settings, and the controlled settings used in the ablation studies. Full details needed to interpret the reported results are included in the main paper or appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

Justification: The paper reports deterministic benchmark metrics and controlled ablations, but it does not report confidence intervals, error bars, or repeated-run statistics for all main experiments. This is mainly due to the computational cost of training the full OCR models, and the paper focuses on consistent evaluation under fixed dataset splits and decoding protocols.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper reports the hardware configuration used for the experiments, including GPU type, memory, and the approximate training or evaluation cost of the main runs. The appendix provides additional information about the computational setting used to reproduce the reported results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research uses existing handwritten text recognition benchmarks and does not involve deceptive studies, unsafe data collection, or deployment on human subjects. The work is presented as an offline recognition model and follows the relevant ethical requirements for reproducible empirical machine learning research.

   Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses potential positive impacts such as more efficient handwritten document recognition and lower-cost document digitization, as well as possible negative impacts related to privacy, surveillance, recognition errors, and uneven performance across writing styles or languages.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release a high-risk generative model, a scraped dataset, or an asset with substantial dual-use misuse risk. The work focuses on handwritten text recognition using existing benchmark datasets.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper cites the original sources of the existing datasets, evaluation protocols, and software components used in the experiments, and describes the applicable access conditions or licenses where available. No existing asset is redistributed beyond the terms under which it is made available.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The submitted supplementary code is documented with instructions for environment setup, data preparation, training, and evaluation. The paper does not introduce a new dataset, and all experiments are based on existing benchmarks.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing experiments or direct research with human subjects. The experiments use existing handwritten text recognition benchmark datasets.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve new human-subject experiments, participant recruitment, or newly collected personal data requiring IRB or equivalent review. The evaluation is conducted on existing benchmark datasets.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method, experiments, and originality of the research do not involve LLMs as an important or non-standard component. Any incidental use of writing or editing tools, if any, does not affect the scientific methodology or experimental results.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.