arXiv:2402.05804v2 [cs.CV] 21 Feb 2024

InkSight: Offline-to-Online Handwriting Conversion by Learning to Read and Write

Blagoj Mitrevski [ⓕ 1] **Arina Rak** [ⓕ 2] **Julian Schnitzler** [ⓕ 2] **Chengkun Li** [ⓕ 2] **Andrii Maksai** [§ 1] **Jesse Berent** [1] **Claudiu Musat** [1]

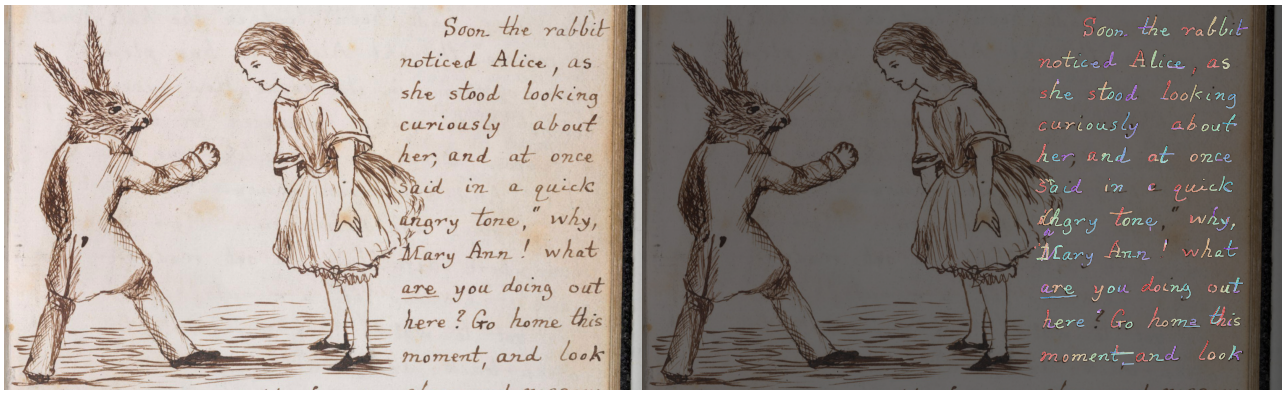

Figure 1: **Results of `InkSight`. Left:** Image of handwriting (offline handwriting), **Right:** output digital inks (online handwriting). In every word, character colors transition from red to purple, following the rainbow sequence; within each stroke, the shade progresses from darker to lighter. More results in Appendix A.

Abstract

Digital note-taking is gaining popularity, offering a durable, editable, and easily indexable way of storing notes in the vectorized form, known as digital ink. However, a substantial gap remains between this way of note-taking and traditional pen-and-paper note-taking, a practice still favored by a vast majority. Our work, `InkSight`, aims to bridge the gap by empowering physical note-takers to effortlessly convert their work (offline handwriting) to digital ink (online handwriting), a process we refer to as **derendering**. Prior research on the topic has focused on the geometric properties of images, resulting in limited generalization beyond their training domains. Our approach combines *reading* and *writing* priors, allowing training a model in the absence of large amounts of paired samples, which are difficult to obtain. To our knowledge, this is the first work that effectively derenders handwritten text in arbitrary photos with diverse visual characteristics and backgrounds. Furthermore, it generalizes beyond its training domain into simple sketches. Our human evaluation reveals that 87% of the samples produced by our model on the challenging HierText dataset are considered as a valid tracing of the input image and 67% look like a pen trajectory traced by a human. Interactive visualizations of 100 word-level model outputs for each of the three public datasets are available in our Hugging Face space. Model release is in progress.

1. Introduction

Handwritten notes have been a cornerstone of information storage for centuries. Today, with the rise of stylus and digital pen technologies, digital inking presents a compelling alternative. This modern approach offers several advantages: enhanced durability, seamless organization and integration with other digital content (images, text, links) or digital assistance. Despite these benefits, many people still prefer traditional handwritten notes over the digital format.

Our work aims to make physical notes, particularly handwritten text, available in the form of digital ink, capturing the stroke-level trajectory details of handwriting. This allows paper note-takers to enjoy the benefits of digital medium without the need to use a stylus. This area has gained significant interest in both academia (Nguyen et al., 2021;

ⓕ Share first authorship, listing order is certified randomized. § Project lead. [1]Google Research [2]EPFL, work done during time as Student Researchers at Google Research. Correspondence to: Andrii Maksai <amaksai@google.com>.

Chen et al., 2022b; Mohamed Moussa et al., 2023) and industry (Not, 2023), with software solutions that digitize handwriting and hardware solutions that require smart pens and/or special paper (Liv, 2024; Roc, 2024; Neo, 2024). Our approach requires only a picture of the handwritten note, without the need for specialized equipment.

Our approach differs from the methods that rely on geometric priors, where gradients, contours, and shapes in an image are utilized to extract writing strokes. Instead, we harness the power of learned *reading* and *writing* priors, where:

- Learned *reading* prior enhances the model's capability in precisely locating and extracting textual elements from the images. This is achieved either through the model's textual understanding ability or aided with external textual input, e.g. from an OCR engine.
- The integration of *writing* prior ensures that the resulting vector representation, the digital ink, closely aligns with the typical human approach of writing in terms of physical dynamics and the order of strokes.

To the best of our knowledge, our work is the first to incorporate these priors, resulting in a robust model that is capable of derendering handwriting across diverse scenarios and appearances, including challenging lighting conditions, noticeable occlusions, etc.

Our model adopts a simple architecture combining the widely popular and readily available ViT (Dosovitskiy et al., 2021) encoder and an mT5 (Xue et al., 2021) encoder-decoder, fostering reproducibility, reusability, and ease of adoption.

To summarize, the major contributions of our work are:

1. We propose the first system to perform **derendering**, transforming arbitrary photos of handwritten text into digital ink.
2. We propose a training and inference setup that works without expensive data collections and scales to arbitrarily large input images.
3. We show that the inks produced by our system are both semantically and geometrically similar to the input images, and are similar to real digital ink data, as measured by both automatic and human evaluations.
4. We show that, due to the learned reading and writing priors, our approach is robust well beyond its training data and works on various types of handwriting, simple sketches, and full pages of notes.

2. Related Work

Pen trajectory recovery has been a task of interest to many researchers due to its utility for tasks that can benefit from stroke order / temporal information such as online handwriting recognition (Lallican et al., 2004; Viard-Gaudin et al., 2005; Zhang et al., 2015; Chan, 2020). Another point of interest is the ability to efficiently store, index, and edit handwritten notes within existing online note taking systems.

Classical approaches commonly consist of domain-specific preprocessing (e.g. noise removal, image binarization, skeletonization), local (sub-)stroke level processing (e.g. identification of junction points) and global aggregation (usually as a graph-based optimization problem) (Jager, 1996; Kato & Yasuhara, 2000; Qiao et al., 2006; Chan, 2020; Doermann et al., 2002). These approaches often rely on the quality of the preprocessing and hand-designed heuristics and do not generalize well to other scripts and domains.

More recent methods use convolutional (Nguyen et al., 2021; Archibald et al., 2021) and/or recurrent (Bhunia et al., 2018; Chen et al., 2022b) neural networks to translate an image into a sequence of coordinates. Mohamed Moussa et al. (2023) use two Transformer models to first encode the sub-strokes of the input and then reorder them, given the encodings. Those methods produce promising results, but focus on the simplified setups of rendered online handwriting and/or single characters.

Line drawing vectorization shares a lot of features with geometric approaches of pen trajectory recovery, but does not include stroke order reconstruction.

Recent techniques for this problem involve solving an optimization task of fitting a set of geometric primitives (e.g. Bezier curves) to match the geometric and/or semantic content of the input image (e.g. Vinker et al. 2022), sometimes relying on differentiable rendering to maintain the inputs and outputs in the image space Mo et al. (2021).

These approaches focus on converting raster images (perfectly aligned, clean, clear backgrounds) to vector ones. Thus they avoid dealing with the realistic photo artifacts related to lighting, noisy backgrounds or photo-taking angles. These artifacts are however unavoidable in a real-world setting.

Dataset availability is limiting the research in trajectory recovery as there are few datasets of images and their corresponding digital inks. These include IRONOFF (Viard-Gaudin et al., 1999), CASIA (Liu et al., 2011), IBM-UB (Shivram et al., 2013) for handwritten text (images there exhibit limited variability being always black-on-white line writing, usually on the same input device) and Sketchy (Sangkloy et al., 2016) for crowd-sourced sketches of photos from 125 object categories.

Combining language models with new modalities is related to this work as we convert the pen trajectory to a

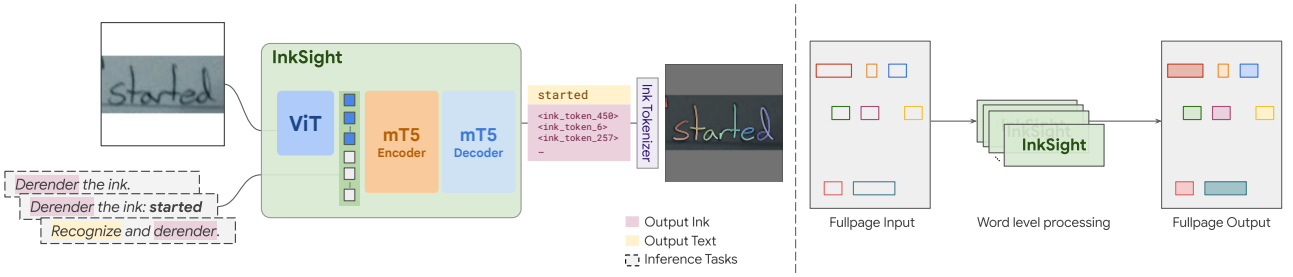

Figure 2: **Left:** Illustration of model inference. Inputs include an image and a prompt specifying the task. Outputs may consist of ink alone or a combination of ink and text, as indicated by matching colors in both the input prompt and the output. Ink tokens are detokenized into digital ink using our proposed ink tokenizer explained in Section 3.3. **Right:** Diagram of the Full Page System.

sequence of coordinates on a fixed size canvas and represent the coordinates with "ink tokens" (more details in Section 3.3). Hence, we consider our approach as extending a Vision-Language Model (VLM) with a new modality.

Recent research in this area has been pointed towards training adapters for merging the pretrained unimodal models (e.g. Alayrac et al. (2022)) or, fully or partially fine-tuning a system with multi-modal tasks (e.g. Chen et al. (2022a)). Works most similar to ours are AudioPaLM (Rubenstein et al., 2023), extending LLM with audio tokens, Painter (Pourreza et al., 2023) using coordinate-based representation of strokes to perform generation and understanding of sketches, and PixelLLM (Xu et al., 2023) using the same representation to localize words in the image caption.

3. Method

While the fundamental concept of `InkSight` appears straightforward – training a model that generates digital ink representations from input images – the practical implementation for arbitrary input images presents two significant challenges: (1) *Limited Supervised Data:* acquiring paired data with corresponding images and ground truth digital ink for supervised training can be expensive and time-consuming and no datasets with sufficient diversity exist for this task. (2) *Scalability to Large Images:* the model must effectively handle potentially arbitrary large input images with varying resolutions and amount of content.

To address the first problem without expensive data collection, we propose a multi-task training setup which combines recognition and derendering tasks. We show that this multi-task training setup enables the model to generalize on derendering tasks with various styles of images as input, and injects the model with both semantic understanding and writing priors of handwritten text. We discuss our data and multi-task training setup in Section 3.2, and ablate the design choices in Section 4.6. The exact model structure, which consists of standard publicly available off-the-shelf components, is discussed in Section 3.1.

The second problem can be addressed by training a model with very high-resolution input images and very long output sequences, but this is computationally prohibitive. Instead, we break down the derendering of a page of notes into three steps: running OCR to extract word-level bounding boxes, derendering each of the words separately and pasting the derendered words back, as shown in Figure 2. We discuss the representation of digital ink, the normalization of the data, and the tokenization scheme we use in Section 3.3. More full page results can be found in Appendix L.

3.1. Vision-Language Model for Digital Ink

In this study, we employ a model architecture inspired by PaLI (Chen et al., 2022a; 2023b;a) which integrates a Vision Transformer (ViT) encoder (Dosovitskiy et al., 2021) with an mT5 encoder-decoder Transformer model (Xue et al., 2021). We provide the task-specific instructions as text, as described in Section 3.2. We use the same set of tokens for both input and output, which contains the individual character tokens from the original mT5 vocabulary and specific tokens reserved for the ink representation (further details in Sec. 3.3).

To initialize the model, we use the pre-trained weights of the ViT encoder. The mT5-based encoder-decoder weights are initialized randomly, which is motivated by our customized token dictionary differing from the one used in mT5's original training. In our training, we freeze the weights of the ViT encoder – we justify this choice empirically in Section 4.6.

3.2. Training Task Mixture

To circumvent the challenge of not having diverse paired image and ink samples as training data, we propose a multi-task training setup comprising two derendering tasks, two recognition tasks, and one hybrid task (shown in Figure 3 and described in Table 1).

We show that this setup helps the model to (1) generalize to the derendering of real photos; (2) learn useful priors which help dealing with occlusions and generating realistic ink; (3) allow using different inference setups which enables, e.g.

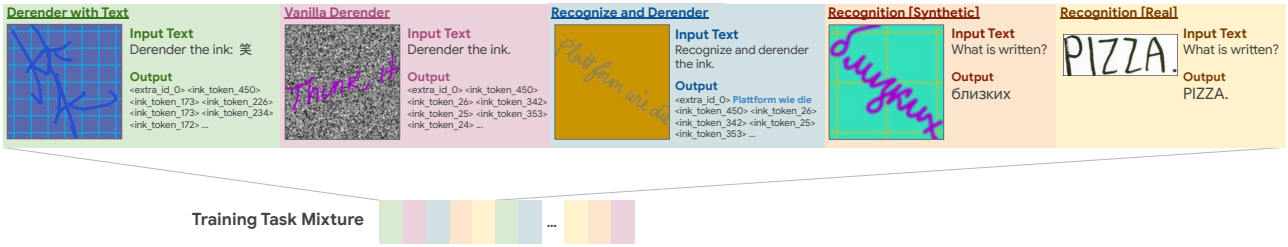

Figure 3: **Illustration of multi-task training mixture.** The training mixture comprises five different task types: two derendering tasks (ink output), two recognition tasks (text output), and one mixed task (text-and-ink output). Each type of task utilizes a task-specific input text, enabling the model to distinguish between tasks during both training and inference.

Table 1: Summary of tasks in our proposed multi-task training.

Tasks	Description
Vanilla Derender	The model receives a synthetic ink image with a conditioning task prompt and outputs ink tokens corresponding to the image content.
Derender with Text	The model receives a synthetic ink image, the target textual content together with a task conditioning prompt, and outputs ink tokens corresponding to the provided text within the image.
Recognition (Syn/Real)	The model receives an ink image (either synthetic or real) and a task conditioning prompt and outputs the recognized textual content. This task uses both synthetic ink images and real images from OCR datasets.
Recognize and Derender	This task combines recognition and derendering. The model receives a synthetic ink image and a prompt and outputs both the recognized text and the corresponding ink tokens.

high-quality text derendering with OCR result (*Derender with Text*) or derendering without textual input for non-textual elements (*Vanilla Derender*).

Training tasks are shuffled and assigned equal appearance probability (20%). By default, during inference, we use *Derender with Text* (this choice is ablated in Section 4.6).

Data augmentation. To narrow the domain gap between the synthetic images of rendered inks and the real photos, we augment the data in tasks that take rendered ink as input. This is done by randomizing the ink angle, color, stroke width, and by adding Gaussian noise and cluttered backgrounds. Examples are provided in Figure 3, more details are given in Appendix E, and Section 4.6 highlights the importance of having the data augmentation.

3.3. Data Representation

Digital ink tokenization. Digital ink is usually represented as a sequence of strokes $I = \{s_1, s_2, \cdots, s_n\}$, where each stroke s_i consists of a sequence of m_i (representing i-th stroke length) coordinate-time triplets, denoted as $s_i = \{(x_i, y_i, t_i)\}_{i=1}^{m_i}$. To enable text decoders like mT5 to generate ink strokes, we first normalize the ink and then tokenize it into a set of discrete tokens from a pre-defined dictionary.

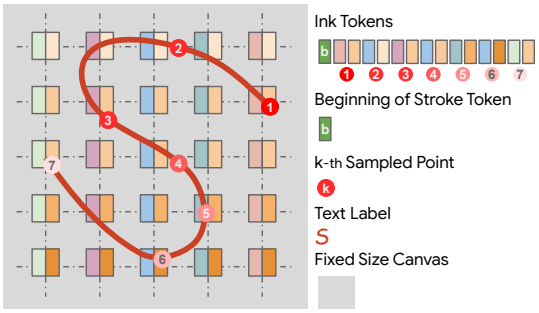

Figure 4: **Illustration of the ink tokenization for a single-stroke ink.** The dark red ink depicts the normalized ink stroke, with numbered circles marking sampled points after time resampling. The color gradient of the sampled points indicates the point order. Each point is represented with two tokens encoding its x and y coordinates. The token sequence for this ink begins with $\mathtt{b}$, signifying the start of the stroke, followed by the tokens for coordinates of sampled points.

Normalizing the ink includes (1) resampling it at a fixed frequency (20 ms) to account for potential differences between input devices; (2) applying Ramer-Douglas-Peucker resampling (Visvalingam & Whyatt, 1990) to reduce the output sequence length while preserving the overall shape; (3) shifting and scaling the ink to place it at the center of fixed size canvas, such that each point in each stroke is within the range of $[0, N]$.

As shown in Figure 4, tokenization of the normalized ink involves prepending a ***beginning of stroke*** token followed by the x and y coordinates of each point inside the stroke, rounded to the nearest integer value. This is done for every stroke inside the digital ink. The total dictionary size is $2N + 3$ with separate set of tokens for x and y, including $N + 1$ possible x and y values and a token indicating a start of the stroke. N controls the tradeoff between rounding error and vocabulary size, and we use $N = 224$ in practice.

Image representation. For the derendering tasks of the training mixture, we render digital inks in the center of an $M \times M$ image with stroke width, color, and background color selected via random augmentation (see Appendix E). For the recognition task training and inference with real images, we apply a similar transformation: the input image is scaled and then centered and padded in black to produce an $M \times M$ image, although we use $M = 224$ in practice, by design there are no constraints to have $N = M$.

Expanding text vocabulary with ink. Our set of tokens contains all individual symbols from the multilingual mT5 tokenizer (Xue et al., 2021) (appx. 20k), with additional $2N + 3$ tokens reserved for the representation of the digital ink, as described above. Removing all multi-symbol tokens allows to reduce the size of the input text embedding and final softmax of the decoder model by $\sim 80\%$, while maintaining the ability to input and output arbitrary text and not affecting the quality of derendering or recognition.

4. Results

In this section, we discuss the training datasets and implementation details, and present the qualitative and quantitative results, followed by an ablation study on training tasks and design choices.

4.1. Datasets

We train our models using two types of datasets: publicly available data and our in-house proprietary data. Below, we detail the public datasets and provide aggregated statistics for the in-house counterparts. Additional details are provided in Appendix B.

OCR training data. As public OCR training data, we use RIMES (Augustin et al., 2006; Grosicki et al., 2009), HierText (Long et al., 2022; 2023), IMGUR5K (Krishnan et al., 2023), ICDAR'15 historical documents (Murdock et al., 2015), and IAM (Marti & Bunke, 1999). We crop out word-level images where possible, acquiring 295 000 samples with text in Latin script.

The in-house dataset consists of images of handwritten and printed text (67% and 33% respectively) written on a diverse set of backgrounds, with 95% of the labels in English.

For data sources that contain word-level segmentation, we extract images of individual words based on their bounding boxes. For others, we found it beneficial to heuristically filter the training data to exclude samples that are too short, too long, or of too low resolution to be rendered on a 224×224 image. The filtering rules ensure that the aspect ratio satisfies ($0.5 <$ width / height < 4.0), and that the image size is at least 25 pixels per side.

Digital ink training data. As public digital ink training data, we use VNOnDB (Nguyen et al., 2018), SCUT-Couch (Li et al., 2008), and DeepWriting (Aksan et al., 2018). For DeepWriting we use the available segmentation information to produce 3 datasets with character-level, word-level, and line-level croppings, and use all 3 for training. For VNOnDB we extract individual words to be used as training data. The public dataset size totals $\sim$2.7M samples.

The in-house multilingual dataset primarily comprises short snippets of text in Mandarin (37%) and Japanese (23%), with other languages contributing under 5%.

Evaluation data. Since there are no available diverse datasets of paired inks and images (as discussed in Section 2), we perform the evaluation on OCR data, and additionally do a small data collection to obtain paired samples.

To automatically assess our models, we evaluate the quality of derendering on the test splits of 3 OCR datasets: IAM (`testset_f`, $\sim$17.6k samples), IMGUR5K ($\sim$23.7k samples), and HierText. For HierText, which was not originally designed around handwritten samples, we apply the same filtering as to the OCR training data, and additionally only consider words marked as handwritten ($\sim$1.3k samples).

Additionally, we have performed a small annotation campaign, asking people to trace $\sim$200 samples from the HierText test set. We use these as the "golden" set to find the reference point in the automated evaluation and as a control group when doing the human evaluation.

4.2. Models

We train 3 variants of our model: **Small-i** ($\sim$340M parameters, **-i** stands for **i**n-house setup) with a ViT B/16 encoder pretrained on JFT-300M (Sun et al., 2017) paired with an mT5-base encoder-decoder; **Small-p** (**-p** stands for **p**ublic setup), built with the same architecture as Small-i, but trained on public datasets and using a publicly available ImageNet-21k (Deng et al., 2009) pretrained ViT B/16 checkpoint; **Large-i** ($\sim$1B parameters) with a ViT L/16 encoder pretrained on JFT-300M, paired with an mT5-large encoder-decoder. All models employ a context length of 1024 for the output, and 128 for the input. Implementation details are provided in Appendix F, and a model card for Small-p in Appendix K.

4.3. Baseline Method

Despite pen trajectory recovery being a popular research direction in the last decades, available comparisons are limited due to the scarcity of open-sourced code, training data, and pre-trained model weights. Appendix C offers a detailed analysis of recent work and their reproducibility. We choose the General Virtual Sketching (GVS) Framework (Mo et al., 2021) as our baseline. Originally trained for sketch and

photo vectorization, GVS offers a valuable reference point despite the domain discrepancy between sketches and handwritten notes. We found that providing the GVS line drawing vectorization model with binarized inputs yields best performance, we compare all 3 available models on original and binarized inputs in Appendix G.

4.4. Qualitative Evaluation

We compare the performance of our models and GVS on 3 public evaluation datasets (mentioned in Section 4.1), in Figure 5. Our models mostly produce results that accurately reflect the text content, disregarding semantically irrelevant background. They can also handle occlusions, highlighting the benefit of the learned *reading* prior, in contrast to GVS, which produces multiple duplicate strokes, and does not distinguish between background and foreground. **Large-i** is able to retain more details and accommodate more diverse image styles.

Figure 5: Comparison between performance of GVS, **Small-i**, **Small-p**, and **Large-i** on 3 public evaluation datasets (IAM, IMGUR5K, HierText from top to bottom), more visualizations in Appendix L.1. In every word, character colors transition from red to purple, following the rainbow sequence; within each stroke, the shade progresses from darker to lighter.

Generalization to sketches. We study performance of our models on out-of-domain samples on simple sketches. We use the *Vanilla Derender* inference mode to obtain the inks. In Figure 6, we observe that our models partly generalize to sketches, but the performance varies significantly depending on the sample and has noticeable artifacts such as missing details or over-focusing on a segment and over-tracing it. They, however, demonstrate robustness to complicated backgrounds (e.g. the lighting of the coffee sample).

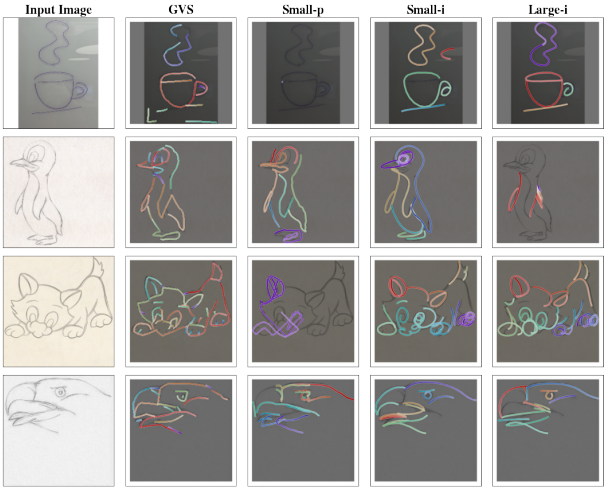

Figure 6: Sketch derendering for **Small-p**, **Small-i**, **Large-i** and GVS. Our models are mostly able to derender simple sketches, however with significant artifacts: missing strokes (e.g the cat), creating cycles (e.g the eye of the penguin for Small-i, the paws of the cat for Large-i) or unnatural strokes (e.g. for Small-p).

4.5. Quantitative Comparison

To support the claims about the performance of our model, we conduct both a human evaluation and an automated evaluation that compare the similarity of our model output to the original image and to real digital inks. There are no established metrics and benchmarks for this problem yet.

4.5.1. Human Evaluation

We performed a human evaluation of the quality of the derendered inks produced by the three variants of our model. We used the "golden" human traced data on the HierText dataset as the control group and the output of our model on these samples as the experiment group. Evaluators were shown the original image alongside a rendered digital ink, which was either model-generated or human-traced (unknown to the evaluators). They were asked to answer two questions: (1) Is the digital ink output a reasonable tracing of the input image? (Answers: Yes, it's a good tracing; It's an okay tracing, but has some small errors; It's a bad tracing, has some major artifacts); (2) Could this digital ink output have been produced by a human? (Answers: Yes; No). We provide a more detailed description of the task and the instructions in Appendix J. The evaluation included 16 individuals familiar with digital ink, but not involved in this research. Each sample was evaluated by 3 raters and aggregated with majority

voting (with inter-rater reliability κ : (1) 0.46, (2) 0.44 for the respective questions).

We demonstrate the results of our evaluation campaign in Figure 7. We observe better performance (good or okay rating, more realistic samples) when both the data and model size are scaled up. The most common imprecisions that lead to choosing "okay tracing" over "good tracing" include: a small number of extra strokes (derendering irrelevant elements present in the image), missing details (e.g. punctuation, a dot over i, j), and unnecessary double strokes (see Appendix J for examples).

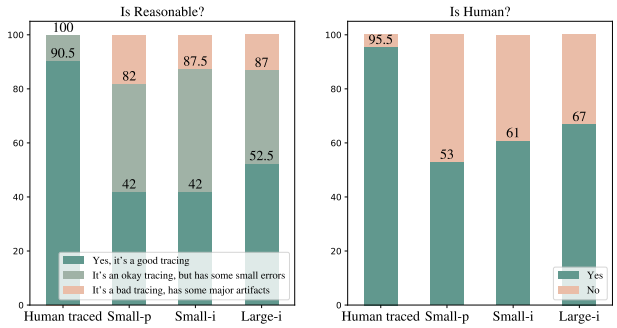

Figure 7: **Human evaluation results for Small-p, Small-i and Large-i** on 200 "golden" samples of HierText dataset.

4.5.2. Automated Evaluation

Automated metrics (described below) are presented in Table 2 and most importantly the ordering using the automated metrics of our models matches the results of the human evaluation.

Table 2: Automated metrics comparison between three variants of our model, GVS, and the "golden" human traced data.

Model	IAM		IMGUR5K		HierText	
	F1	Acc.	F1	Acc.	F1	Acc.
Small-p	0.65	0.60	0.47	0.33	0.53	0.37
Small-i	0.65	0.59	0.51	0.33	0.61	0.44
Large-i	0.66	0.58	0.51	0.32	0.61	0.46
GVS	0.69	0.02	0.51	0.01	0.55	0.01
Human*	–	–	–	–	0.64	0.74

* Computed on the human-traced subset of HierText data.

Similarity to the input image. To capture how similar the output of our model is to the input image semantically and geometrically, we compute the Character Level F1 score by following the protocol of the Robust Reading Challenge (Rob, 2023). This measure captures both geometry and semantics by counting the character bounding boxes found by an OCR model (in the original input image and in the output of the model rendered as an image) that both overlap significantly and match content-wise. To obtain those character bounding boxes, we use the publicly available API (Clo, 2024).

We see that our models perform similarly to GVS on simpler black-on-white IAM dataset, but outperforms it on HierText which has more diverse background. We also note that the OCR engine used is not perfect, and due to errors in the OCR model and its sensitivity to line width, the value of this metric is 0.64 for the "golden" human traced data on HierText. Therefore, it can only be used to meaningfully distinguish models as long as their F1 score is fairly low.

Similarity to the real digital ink data. To assess the semantic consistency and geometric properties of our generated digital inks, we compute recognition Exact Match Accuracy with the state-of-the-art online handwriting recognizer (Carbune et al., 2020) trained on real digital ink data.

With results shown in Table 2, our models show superior accuracy over GVS (on which the online handwriting recognizer is struggling due to unnatural stroke number, length and order). We also observed that while the three model variants perform similarly on the IAM and IMGUR5K datasets, the **Large** variant exhibits better performance compared to the **Small** variants on the HierText dataset which has greater text source diversity. This aligns with our human evaluation.

Table 3: Online handwriting recognition results on IAMOnDB test set for Small-i trained on real digital inks (IAMOnDB train set), derendered digital inks (from IAM train set) and their combination.

	IAMOnDB	**IAM derendered**	**IAMOnDB + IAM derendered**
CER	6.1	7.8	4.6

To further confirm the similarity of real and derendered digital inks in terms of low-level feature statistics, we train an online handwriting recognizer on inks derendered from the IAM training set. We measure the performance on the `testset_f` of IAMOnDB (Liwicki & Bunke, 2005) and compare it with the model trained on IAMOnDB, with the commonly used Character Error Rate metric (CER). As shown in Table 2, model trained on derendered inks only has CER performance that is worse, but not far off from the one trained on real data. Furthermore, the combination of derendered IAM inks and real IAMOnDB inks allows to acquire a more diverse training set, resulting in significantly lower CER. We verify that this observation holds for **Small-p** and **Large-i** (see Appendix H). We attribute the relatively far from state-of-the-art CER of 6.1 to the small size of the training set. The details of the online handwriting recognizer we train are given in Appendix H.

4.6. Ablation Studies

This section examines the performance variation across inference tasks and ablates the impact of augmentation, data mixture, and training strategies on the **Small-i** model's performance. Conclusions and numerical results of the ablation study are presented in Table 4 and summarized below. For more details, see Appendix D.

Table 4: Ablation studies on **Small-i**. The first row shows the original performance (with inference mode *Derender with Text*). Subsequent rows display results for the *Vanilla Derender* and *Recognize and Derender* tasks (Figure 3), followed by the impact of various design choices on performance.

Setup	IAM		IMGUR5K		HierText	
	F1	Acc.	F1	Acc.	F1	Acc.
Small-i†	0.66±0.07	0.59±0.01	0.51±0.09	0.33±0.02	0.61±0.07	0.45±0.01
Vanilla	0.61↓	0.53↓	0.44↓	0.28↓	0.53↓	0.35↓
R+D	0.62↓	0.57↓	0.46↓	0.32	0.57↓	0.42↓
Remove						
data aug*	0.42↓	0.33↓	0.21↓	0.09↓	0.23↓	0.13↓
syn rec	0.58↓	0.50↓	0.50↓	0.25↓	0.56↓	0.38↓
real rec	0.64↓	0.50↓	0.55↑	0.19↓	0.59↓	0.36↓
all rec	0.65	0.51↓	0.55↑	0.22↓	0.61	0.38↓
fr ViT†	0.65±0.13	0.53±0.06↓	0.43±0.24	0.31±0.06	0.59±0.15	0.41±0.05↓

* Digital ink rendered as fixed-width white strokes on black background.
† The results are aggregated from three random initializations, all other results are acquired with the best performing initializations with the same three seeds.

Inference task matters (Row 2-3). We compare the performance between three types of inference tasks, we find *Derender with Text* is better than both *Vanilla Derender* and *Recognize and Derender* (R+D) on all three datasets, we attribute this difference to the fact that it generates more semantically consistent inks than the other two. Figure 8 shows a collection of derendered inks from **Small-i** where each input image has ambiguous or difficult-to-read characters. Inference with *Vanilla Derender* gives results that capture the overall geometry of the input image, but can lack textual understanding. *Derender with Text* gives results that are consistent with the text input to the model provided by OCR, while *Recognize and Derender* gives results that are consistent with the model's own recognition. More examples in Appendix D.2.

The necessity of data augmentation (Row 5). Removing the data augmentation leads to significantly worse performance across all metrics on all datasets. While data augmentation (as shown in Figure 3) does not visually align rendered inks with real ink photos, it diversifies the rendered ink distribution and we find that it is essential for the model to perform valid derendering on real images with our training setup.

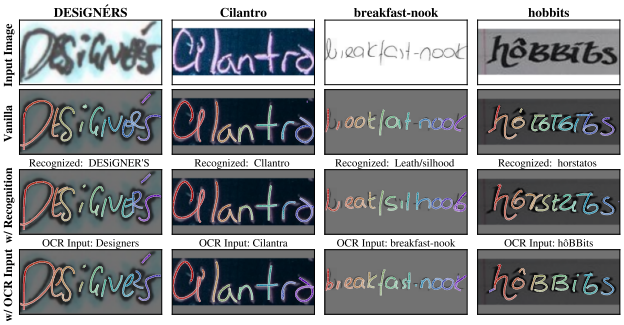

Figure 8: Varied digital ink outputs from **Small-i** inference tasks on samples with ambiguous transcription. Column titles correspond to the ground truth labels. Image titles correspond to either the recognition result from our model or an OCR system.

Recognition tasks improve derendering quality (Row 6-8). Removing recognition tasks from the multi-task training mixture, while maintaining the relative ratios of other tasks, notably reduces overall performance across all evaluation datasets, particularly impacting accuracy which reflects the model's semantic understanding ability.

Impact of frozen ViT in multi-task training. (Last row). Unfreezing the Vision Transformer (ViT) in multi-task training typically incurs training instability, as evidenced by significant variance in model evaluations and an increased tendency to misinterpret background noise as textual content (see Appendix D). The observed increase in F1 scores can be attributed to the model interpreting more background noise as strokes, which leads to digital inks that closely mimic the style of the input images, resulting in a higher likelihood of overlapping bounding boxes in F1 calculations. More details in Appendix D.1.

5. Conclusion and Future Work

In this work we present the first approach to converting photos of handwriting into digital ink. We propose a training setup that works without paired training data, which can be difficult to acquire. We show that our method is robust to a variety of input conditions, can work on full handwritten notes, and generalizes to out-of-domain sketches to some extent. Furthermore, our approach does not require complex modelling and can be constructed from standard building blocks. Future work may address the main limitation of our model which is the need to have a page segmentation to run derendering on the individual words identified by the segmentation.

6. Impact Statements

6.1. Ethical Considerations

Proposed model can not be used for open-ended generation. We verify that proposed models are unable to per-

form open-ended ink generation. We provide random texts in task prompts of task *Derender with Text* (in Figure 3) with input images either empty or contain handwritings that do not match the label. We observe that the generated inks do not match the labels for 100 cases inspected, signifying that the model is unable to generate the prompted text that is not present in the input image. More details in Appendix L.2.

6.2. Future Societal Consequences

Our work could allow access to the digital ink underlying the physical notes, potentially enabling the training of better online handwriting recognizers for languages that are historically low-resource in the digital ink domain.

7. Acknowledgement

The authors thank Leandro Kieliger for the help on synthetic rendered ink rendering, feedback on initial versions of the paper, Philippe Schlattner on human evaluation protocol building, Anastasia Fadeeva, Mircea Trăichioiu for the multiple discussions on weekly meetings, Efi Kokiopoulou, Diego Antognini, Henry Rowley, Reeve Ingle, Manuel Drazyk for feedback on initial versions of the paper, Sebastian Goodman, Jialin Wu for the help on implementing deterministic training and PaLI related questions, Xiao Wang on ViT related questions, Tom Duerig and Tomáš Ižo for leadership support.

References

Notability Ink to Image conversion. https://support.gingerlabs.com/hc/en-us/articles/5044440428570-Image-to-Ink-Conversion, 2023. [Online; accessed 10-Jan-2024].

Hierarchical Text: Challenge on Unified OCR and Layout Analysis. https://rrc.cvc.uab.es/?ch=18&com=tasks, 2023. [Online; accessed 10-Jan-2024].

Google Cloud Vision Handwriting Text Detection API. https://cloud.google.com/vision/docs/handwriting, 2024. [Online; accessed 10-Jan-2024].

LiveScribe. https://us.livescribe.com/, 2024. [Online; accessed 10-Jan-2024].

NeoSmartPen. https://neosmartpen.com/, 2024. [Online; accessed 10-Jan-2024].

Rocketbook. https://getrocketbook.com/, 2024. [Online; accessed 10-Jan-2024].

Aksan, E., Pece, F., and Hilliges, O. Deepwriting: Making digital ink editable via deep generative modeling. In *Proceedings of the 2018 CHI conference on human factors in computing systems*, pp. 1–14, 2018.

Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022.

Alwajih, F., Badr, E., and Abdou, S. Transformer-based models for arabic online handwriting recognition. *International Journal of Advanced Computer Science and Applications*, 13(5), 2022.

Archibald, T., Poggemann, M., Chan, A., and Martinez, T. Trace: a differentiable approach to line-level stroke recovery for offline handwritten text. In *Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part III 16*, pp. 414–429. Springer, 2021.

Augustin, E., Carré, M., Grosicki, E., Brodin, J.-M., Geoffrois, E., and Prêteux, F. Rimes evaluation campaign for handwritten mail processing. In *International Workshop on Frontiers in Handwriting Recognition (IWFHR'06),*, pp. 231–235, 2006.

Bhunia, A. K., Bhowmick, A., Bhunia, A. K., Konwer, A., Banerjee, P., Roy, P. P., and Pal, U. Handwriting trajectory recovery using end-to-end deep encoder-decoder network. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pp. 3639–3644. IEEE, 2018.

Bhunia, A. K., Chowdhury, P. N., Yang, Y., Hospedales, T. M., Xiang, T., and Song, Y.-Z. Vectorization and rasterization: Self-supervised learning for sketch and handwriting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5672–5681, 2021.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., et al. Jax: composable transformations of python+ numpy programs. 2018.

Carbune, V., Gonnet, P., Deselaers, T., Rowley, H. A., Daryin, A., Calvo, M., Wang, L.-L., Keysers, D., Feuz, S., and Gervais, P. Fast multi-language lstm-based online handwriting recognition. *International Journal on Document Analysis and Recognition (IJDAR)*, 23(2):89–102, 2020.

Chan, C. Stroke extraction for offline handwritten mathematical expression recognition. *IEEE Access*, 8:61565–61575, 2020.

Chen, X., Wang, X., Changpinyo, S., Piergiovanni, A., Padlewski, P., Salz, D., Goodman, S., Grycner, A., Mustafa, B., Beyer, L., et al. Pali: A jointly-scaled multilingual language-image model. *arXiv preprint arXiv:2209.06794*, 2022a.

Chen, X., Djolonga, J., Padlewski, P., Mustafa, B., Changpinyo, S., Wu, J., Ruiz, C. R., Goodman, S., Wang, X., Tay, Y., et al. Pali-x: On scaling up a multilingual vision and language model. *arXiv preprint arXiv:2305.18565*, 2023a.

Chen, X., Wang, X., Beyer, L., Kolesnikov, A., Wu, J., Voigtlaender, P., Mustafa, B., Goodman, S., Alabdulmohsin, I., Padlewski, P., et al. Pali-3 vision language models: Smaller, faster, stronger. *arXiv preprint arXiv:2310.09199*, 2023b.

Chen, Z., Yang, D., Liang, J., Liu, X., Wang, Y., Peng, Z., and Huang, S. Complex handwriting trajectory recovery: Evaluation metrics and algorithm. In *Proceedings of the asian conference on computer vision*, pp. 1060–1076, 2022b.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Doermann, D., Intrator, N., Rivin, E., and Steinherz, T. Hidden loop recovery for handwriting recognition. In *Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition*, pp. 375–380. IEEE, 2002.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Etter, D., Rawls, S., Carpenter, C., and Sell, G. A synthetic recipe for ocr. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pp. 864–869, 2019. doi: 10.1109/ICDAR.2019.00143.

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

Heek, J., Levskaya, A., Oliver, A., Ritter, M., Rondepierre, B., Steiner, A., and van Zee, M. Flax: A neural network library and ecosystem for JAX, 2023. URL http://github.com/google/flax.

Jager, S. Recovering writing traces in off-line handwriting recognition: Using a global optimization technique. In *Proceedings of 13th international conference on pattern recognition*, volume 3, pp. 150–154. IEEE, 1996.

Kato, Y. and Yasuhara, M. Recovery of drawing order from single-stroke handwriting images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(9):938–949, 2000.

Krishnan, P., Kovvuri, R., Pang, G., Vassilev, B., and Hassner, T. Textstylebrush: Transfer of text aesthetics from a single example. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

Lallican, P., Viard-Gaudin, C., and Knerr, S. From off-line to on-line handwriting recognition. In *EPRINTS-BOOK-TITLE*. 2004.

Li, Y., Jin, L., Zhu, X., and Long, T. Scut-couch2008: A comprehensive online unconstrained chinese handwriting dataset. *ICFHR*, 2008:165–170, 2008.

Liu, C.-L., Yin, F., Wang, D.-H., and Wang, Q.-F. Casia online and offline chinese handwriting databases. In *2011 international conference on document analysis and recognition*, pp. 37–41. IEEE, 2011.

Liwicki, M. and Bunke, H. Iam-ondb - an on-line english sentence database acquired from handwritten text on a whiteboard. In *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)*, pp. 956–961 Vol. 2, 2005. doi: 10.1109/ICDAR.2005.132.

Long, S., Qin, S., Panteleev, D., Bissacco, A., Fujii, Y., and Raptis, M. Towards end-to-end unified scene text detection and layout analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1049–1059, 2022.

Long, S., Qin, S., Panteleev, D., Bissacco, A., Fujii, Y., and Raptis, M. Icdar 2023 competition on hierarchical text detection and recognition. *arXiv preprint arXiv:2305.09750*, 2023.

Marti, U.-V. and Bunke, H. A full english sentence database for off-line handwriting recognition. In *Proceedings of the Fifth International Conference on Document Analysis and Recognition. ICDAR'99 (Cat. No. PR00318)*, pp. 705–708. IEEE, 1999.

Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., Spitzer, E., Raji, I. D., and Gebru, T. Model cards for model reporting. In *Proceedings of the conference on fairness, accountability, and transparency*, pp. 220–229, 2019.

Mo, H., Simo-Serra, E., Gao, C., Zou, C., and Wang, R. General virtual sketching framework for vector line art. *ACM Transactions on Graphics (TOG)*, 40(4):1–14, 2021.

Mohamed Moussa, E., Lelore, T., and Mouchère, H. Set, sort! a novel sub-stroke level transformers for offline handwriting to online conversion. In *International Conference on Document Analysis and Recognition*, pp. 81–97. Springer, 2023.

Murdock, M., Reid, S., Hamilton, B., and Reese, J. Icdar 2015 competition on text line detection in historical documents. In *2015 13th international conference on document analysis and recognition (ICDAR)*, pp. 1171–1175. IEEE, 2015.

Nguyen, H. T., Nguyen, C. T., and Nakagawa, M. Icfhr 2018–competition on vietnamese online handwritten text recognition using hands-vnondb (vohtr2018). In *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp. 494–499. IEEE, 2018.

Nguyen, H. T., Nakamura, T., Nguyen, C. T., and Nakawaga, M. Online trajectory recovery from offline handwritten japanese kanji characters of multiple strokes. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 8320–8327. IEEE, 2021.

Pourreza, R., Bhattacharyya, A., Panchal, S., Lee, M., Madan, P., and Memisevic, R. Painter: Teaching autoregressive language models to draw sketches. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pp. 305–314, October 2023.

Qiao, Y., Nishiara, M., and Yasuhara, M. A framework toward restoration of writing order from single-stroked handwriting image. *IEEE transactions on pattern analysis and machine intelligence*, 28(11):1724–1737, 2006.

Roberts, A., Chung, H. W., Mishra, G., Levskaya, A., Bradbury, J., Andor, D., Narang, S., Lester, B., Gaffney, C., Mohiuddin, A., et al. Scaling up models and data with t5x and seqio. *Journal of Machine Learning Research*, 24(377):1–8, 2023.

Rubenstein, P. K., Asawaroengchai, C., Nguyen, D. D., Bapna, A., Borsos, Z., Quitry, F. d. C., Chen, P., Badawy, D. E., Han, W., Kharitonov, E., et al. Audiopalm: A large language model that can speak and listen. *arXiv preprint arXiv:2306.12925*, 2023.

Sangkloy, P., Burnell, N., Ham, C., and Hays, J. The sketchy database: learning to retrieve badly drawn bunnies. *ACM Transactions on Graphics (TOG)*, 35(4):1–12, 2016.

Shazeer, N. and Stern, M. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pp. 4596–4604. PMLR, 2018.

Shivram, A., Ramaiah, C., Setlur, S., and Govindaraju, V. Ibm_ub_1: A dual mode unconstrained english handwriting dataset. In *2013 12th International Conference on Document Analysis and Recognition*, pp. 13–17. IEEE, 2013.

Sumi, T., Iwana, B. K., Hayashi, H., and Uchida, S. Modality conversion of handwritten patterns by cross variational autoencoders. In *2019 international conference on document analysis and recognition (ICDAR)*, pp. 407–412. IEEE, 2019.

Sun, C., Shrivastava, A., Singh, S., and Gupta, A. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pp. 843–852, 2017.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Viard-Gaudin, C., Lallican, P. M., Knerr, S., and Binter, P. The ireste on/off (ironoff) dual handwriting database. In *Proceedings of the Fifth International Conference on Document Analysis and Recognition. ICDAR'99 (Cat. No. PR00318)*, pp. 455–458. IEEE, 1999.

Viard-Gaudin, C., Lallican, P.-M., and Knerr, S. Recognition-directed recovering of temporal information from handwriting images. *Pattern Recognition Letters*, 26(16):2537–2548, 2005.

Vinker, Y., Pajouheshgar, E., Bo, J. Y., Bachmann, R. C., Bermano, A. H., Cohen-Or, D., Zamir, A., and Shamir, A. Clipasso: Semantically-aware object sketching. *ACM Transactions on Graphics (TOG)*, 41(4):1–11, 2022.

Visvalingam, M. and Whyatt, J. D. The douglas-peucker algorithm for line simplification: re-evaluation through visualization. In *Computer graphics forum*, volume 9, pp. 213–225. Wiley Online Library, 1990.

Xu, J., Zhou, X., Yan, S., Gu, X., Arnab, A., Sun, C., Wang, X., and Schmid, C. Pixel Aligned Language Models. *arXiv preprint arXiv: 2312.09237*, 2023.

Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., Barua, A., and Raffel, C. mT5: A massively multilingual pre-trained text-to-text transformer. In Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y. (eds.), *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 483–498, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.41. URL https://aclanthology.org/2021.naacl-main.41.

Zhai, X., Wang, X., Mustafa, B., Steiner, A., Keysers, D., Kolesnikov, A., and Beyer, L. Lit: Zero-shot transfer with locked-image text tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18123–18133, 2022.

Zhang, X., Wang, M., Wang, L., Huo, Q., and Li, H. Building handwriting recognizers by leveraging skeletons of both offline and online samples. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pp. 406–410. IEEE, 2015.

A. Full-page Results

We show the full-page results on samples that resemble real-life inputs (mostly from Unsplash with keyword search for "handwriting", and others collected by the authors with consent from the writers) produced by three variants of our models.

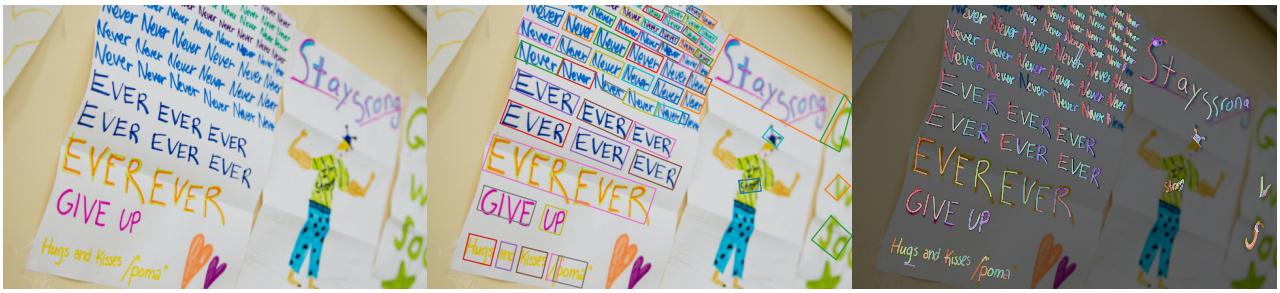

Figure 9: **Large-i** full page results of handwritten notes in a real-life scenario, credit: Unsplash.

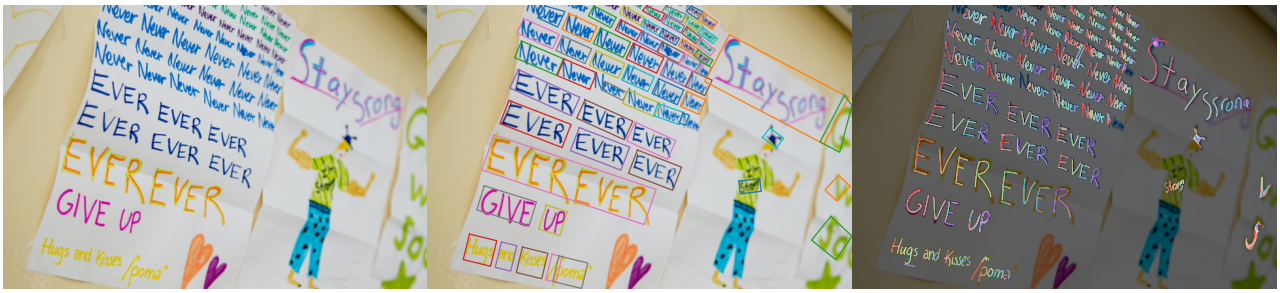

Figure 10: **Small-i** full page results of handwritten notes in a real-life scenario, credit: Unsplash.

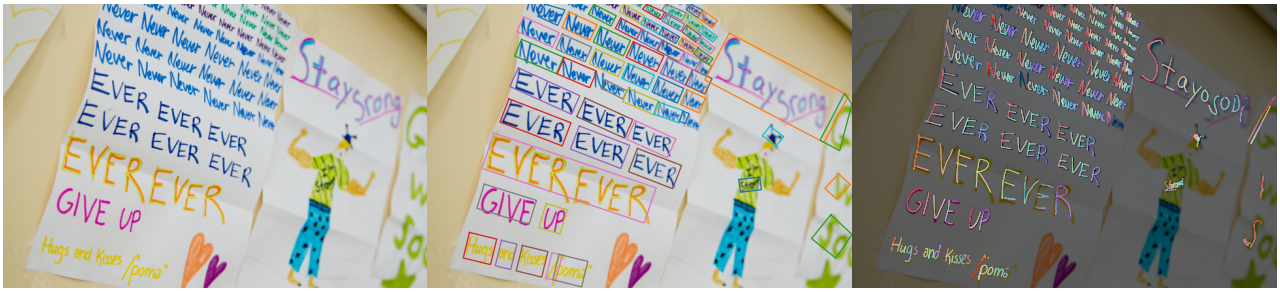

Figure 11: **Small-p** full page results of handwritten notes in a real-life scenario, credit: Unsplash.

Figure 12: **Large-i** full page results of French sample. credit: Unsplash

Figure 13: **Small-i** full page results of French sample. credit: Unsplash

Figure 14: **Small-p** full page results of French sample. credit: Unsplash

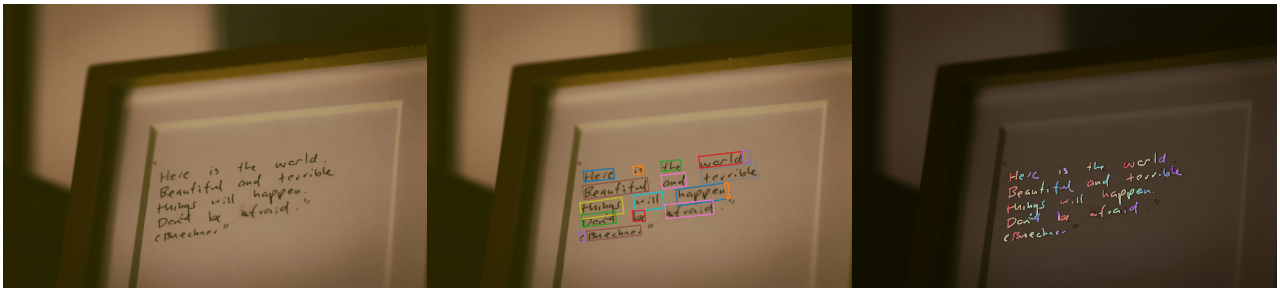

Figure 15: **Large-i** full page results of handwritten notes in a real-life scenario with low resolution, credit: Unsplash.

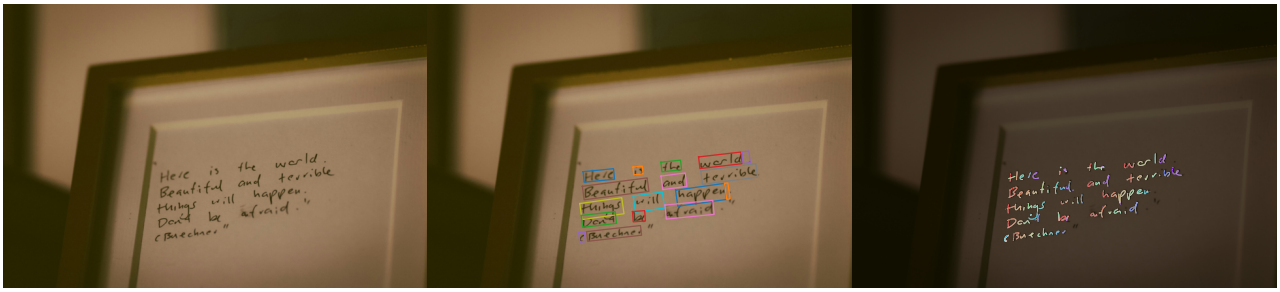

Figure 16: **Small-i** full page results of handwritten notes in a real-life scenario with low resolution, credit: Unsplash.

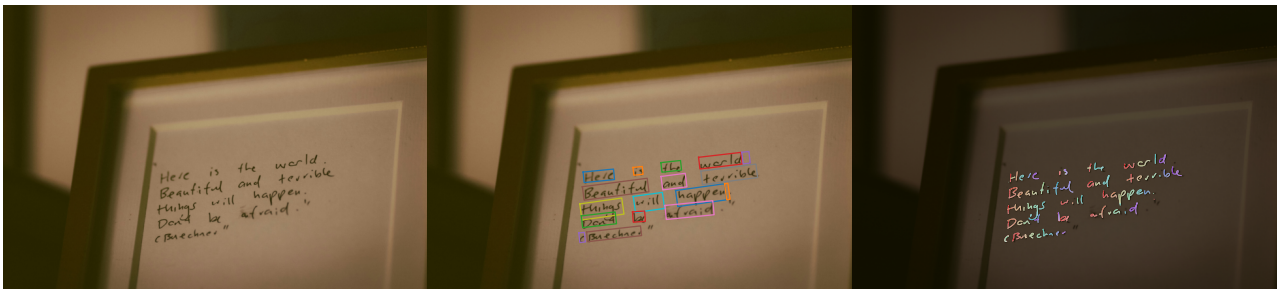

Figure 17: **Small-p** full page results of handwritten notes in a real-life scenario with low resolution, credit: Unsplash.

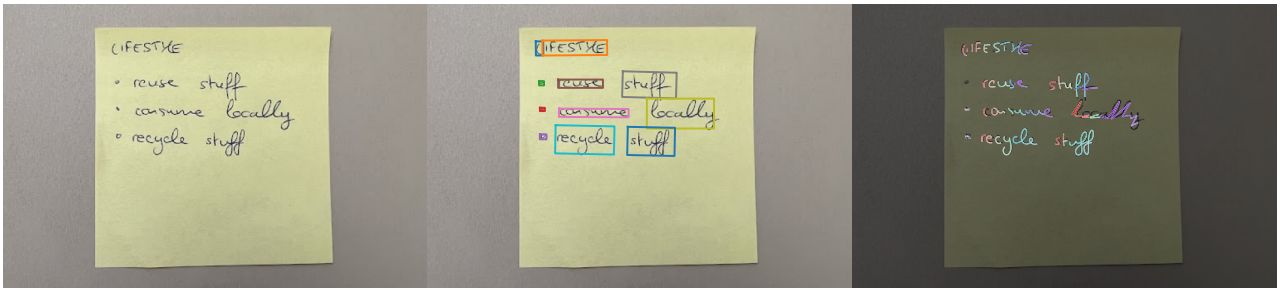

Figure 18: **Large-i** full page results of a Post-It note.

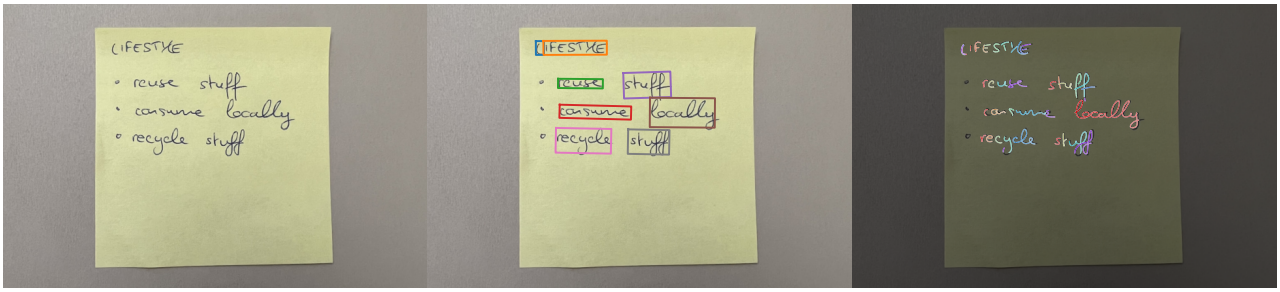

Figure 19: **Small-i** full page results of a Post-It note.

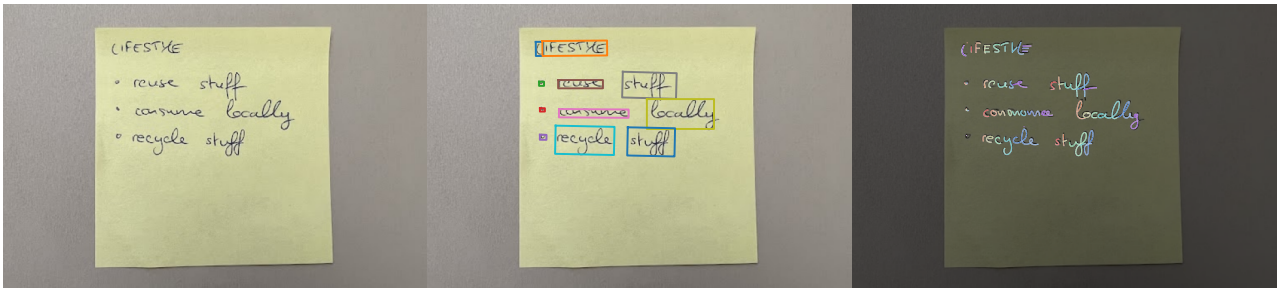

Figure 20: **Small-p** full page results of a Post-It note.

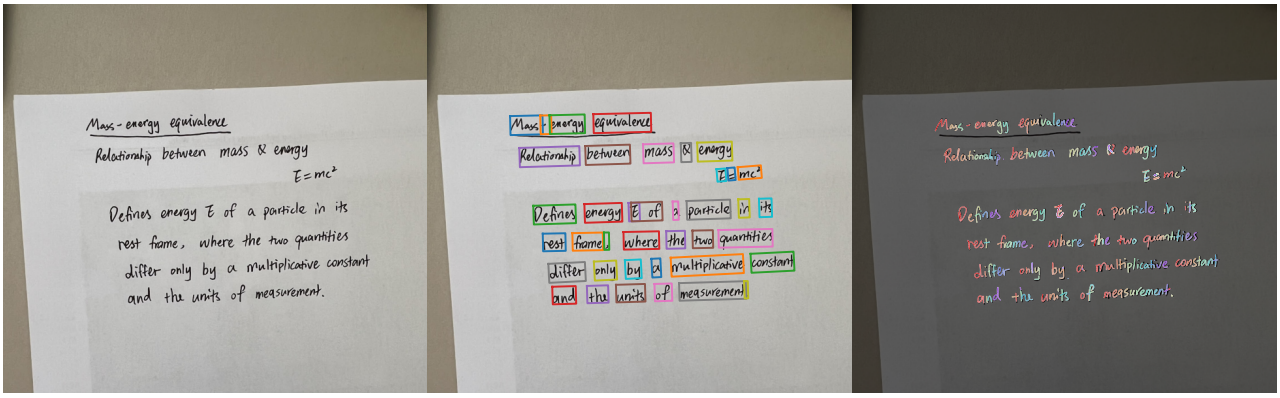

Figure 21: **Large-i** full page results of handwritten notes of mass-energy equivalence.

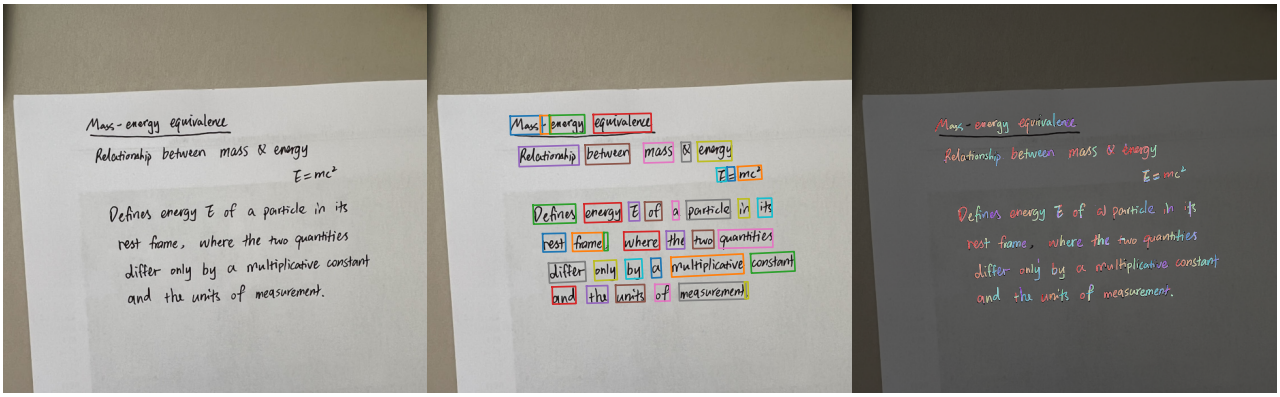

Figure 22: **Small-i** full page results of handwritten notes of mass-energy equivalence.

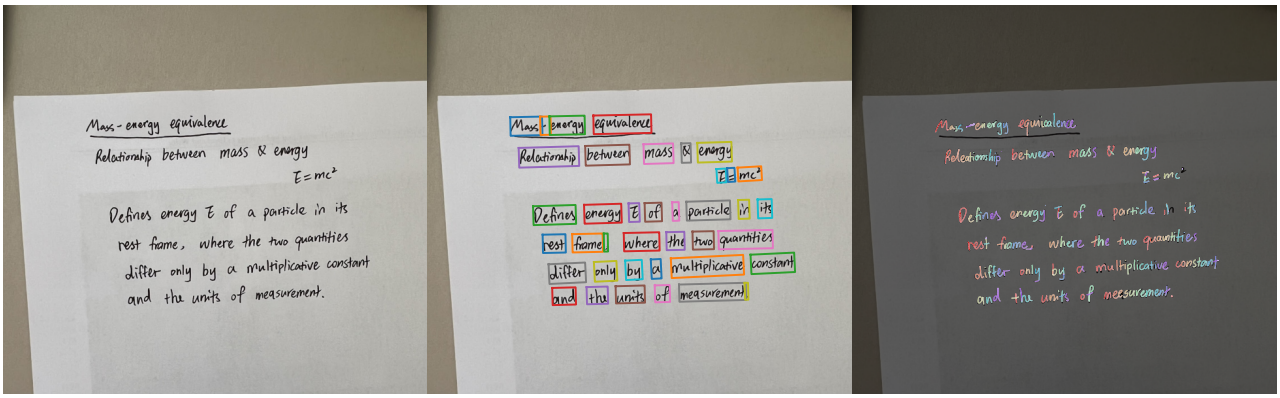

Figure 23: **Small-p** full page results of handwritten notes of mass-energy equivalence.

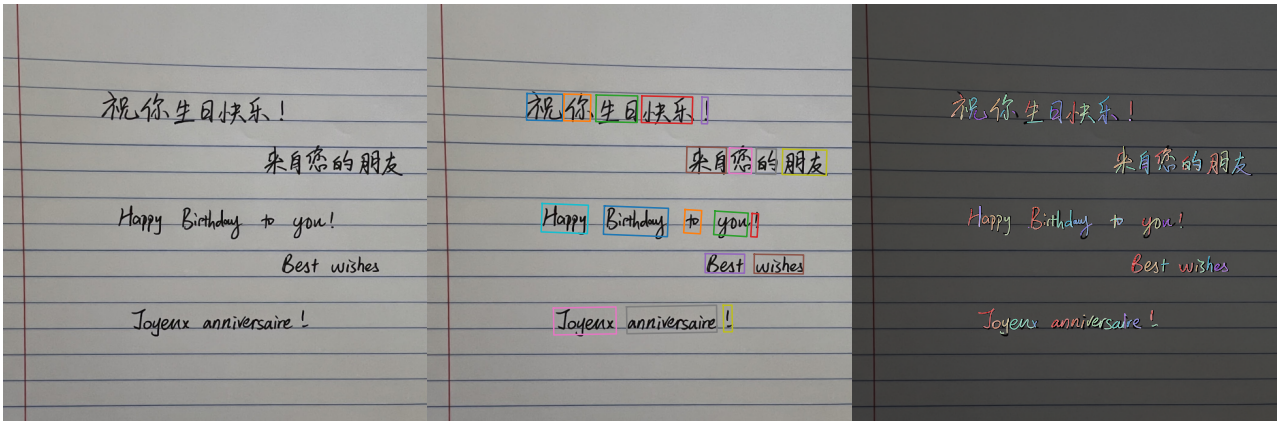

Figure 24: **Large-i** full page results of multilingual handwritten notes.

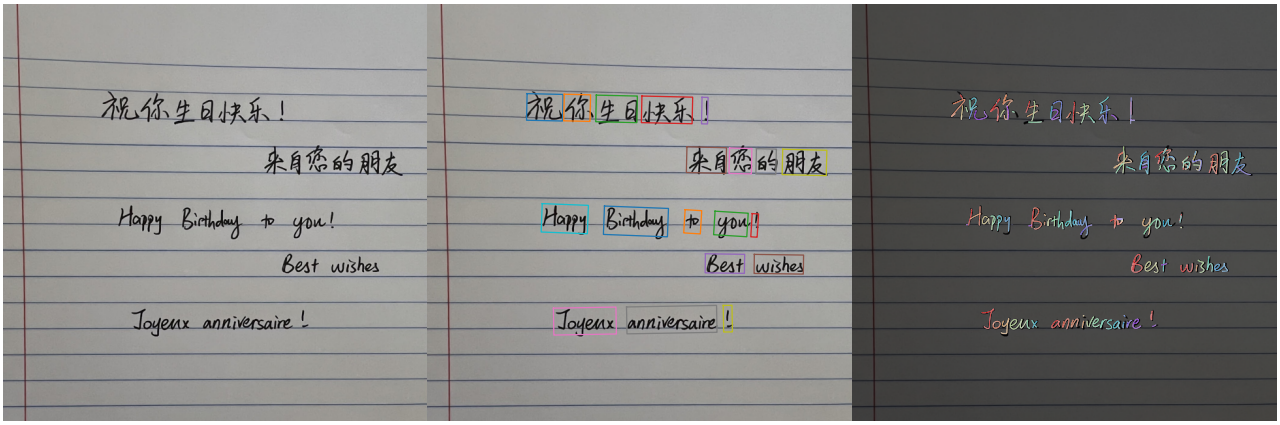

Figure 25: **Small-i** full page results of multilingual handwritten notes.

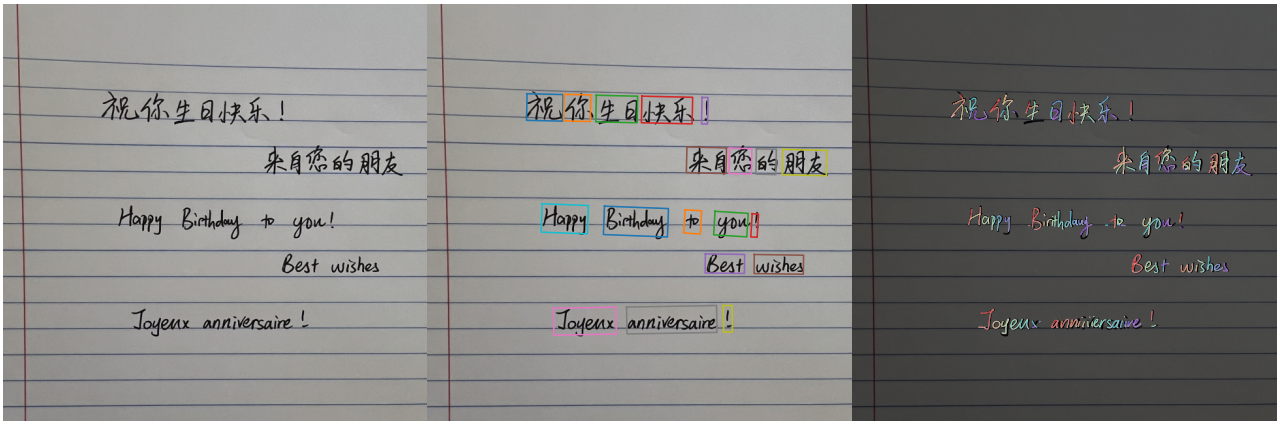

Figure 26: **Small-p** full page results of multilingual handwritten notes.

Figure 27: **Large-i** full page results of OOD sample.

Figure 28: **Small-i** full page results of OOD sample.

Figure 29: **Small-p** full page results of OOD sample.

Figure 30: **Large-i** full page results of OOD sample, credit: Unsplash.

Figure 31: **Small-i** full page results of OOD sample, credit: Unsplash.

Figure 32: **Small-p** full page results of OOD sample, credit: Unsplash.

B. Data statistics and Preprocessing

We present the language distribution of the digital ink datasets used to train our in-house (**-i** suffix) and publicly available (**-p** suffix) models in Figures. 33, 34 and specific datasets and corresponding number of samples used in training our public model in Table 5

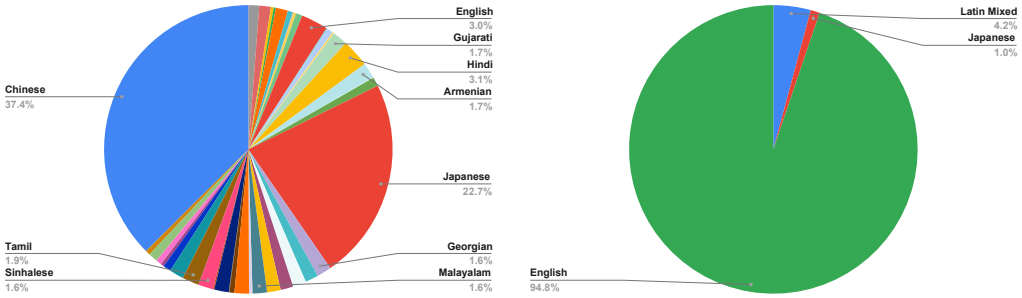

Figure 33: In-house Datasets Language Distributions. **Left:** Digital inks, **Right:** OCR for Recognition. These datasets are used to train both in-house models **Small-i** and **Large-i**.

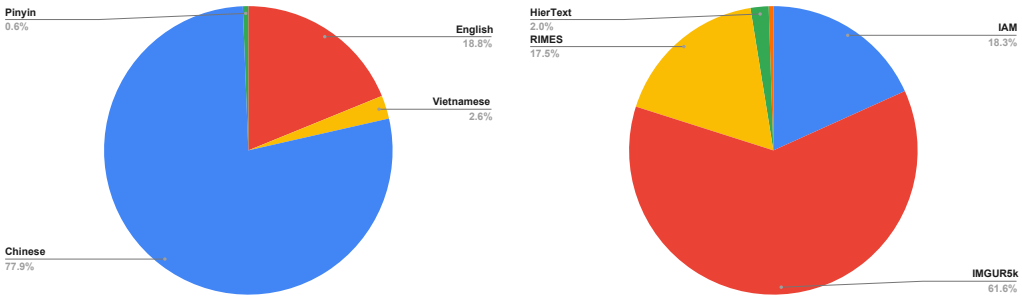

Figure 34: Public Datasets. Language distribution of public Digital ink datasets used to train our public models (on the left) and public OCR datasets used to train our public available model **Small-p** (on the right).

Table 5: Public training datasets used for training **small-p**.

Task Type	Dataset	Number of Samples
Derendering	DeepWriting (words)	89,565
	DeepWriting (lines)	33,933
	DeepWriting (characters)	359,643
	VNOnDB	66,991
	SCUT-COUCH Chinese characters	1,998,784
	SCUT-COUCH Chinese pinyin	156,53
OCR	IAM word-level (train)	53,839
	IMGUR5k (train)	181,792
	RIMES word-level (train)	51,738
	HierText (train)	5,978
	ICDAR-2015 (train)	1,535

C. Survey of Derendering Models

In this section we provide a small survey of recent published pen trajectory recovery approaches and show that none of them release sufficient materials to reproduce their work with a reasonable amount of effort. We illustrate this with Table 6.

Table 6: Existing pen trajectory recovery approaches and their reproducibility.

Work	Has code	Has data	Has model weights	Year
Mohamed Moussa et al. (2023)	✗	✓	✗	2023
Chen et al. (2022b)	✗ repository missing core elements: data preprocessing, outdated/incompetable config files	✓	✗	2022
Archibald et al. (2021)	✗	✓	✗	2021
Bhunia et al. (2021)	✗ repository missing core elements: training scripts, data preprocessing	✓	✗	2021
Nguyen et al. (2021)	✗	✓	✗	2021
Sumi et al. (2019)	✗	✓	✗	2019
Bhunia et al. (2018)	✓	✗	✗	2018

D. Further Details of Ablation Studies

D.1. Frozen ViT helps training stability

We investigated the impact of freezing the vision encoder during training on model stability and derendering performance. To assess training stability, we analyzed the ratio of empty ink predictions across training steps on the "golden" human traced dataset (detailed in Section 4.5.1). Empty inks typically occur when the model confuses tasks, mistakenly outputting text instead of ink or both (different tasks depicted in Figure 3). Our analysis revealed significantly greater variance between runs with an unfrozen setup compared to the consistent learning observed with a frozen vision encoder (Figure 35).

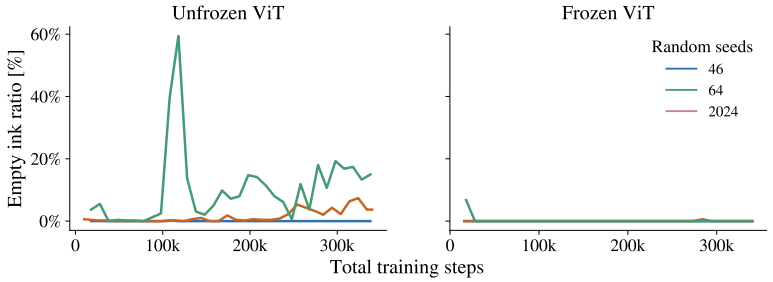

Figure 35: **Comparison between frozen and unfrozen ViT Small-i trainings.** Comparison is executed with 3 random seeds. **Left:** ratio of empty ink outputs for unfrozen ViT multi-task training setup. **Right:** ratio of empty ink outputs for the frozen ViT multi-task training setup, both use inference task *Derender with Text*.

Furthermore, in terms of final model derendering performance, we identified that certain model initializations (controlled by seeds) within the unfrozen setup could also converge to functional models with comparable abilities to those trained with a frozen setup. These models demonstrate superior preservation of visual ink details but exhibit increased sensitivity to background noise. We present a visual comparison of their output characteristics in Figure 36.

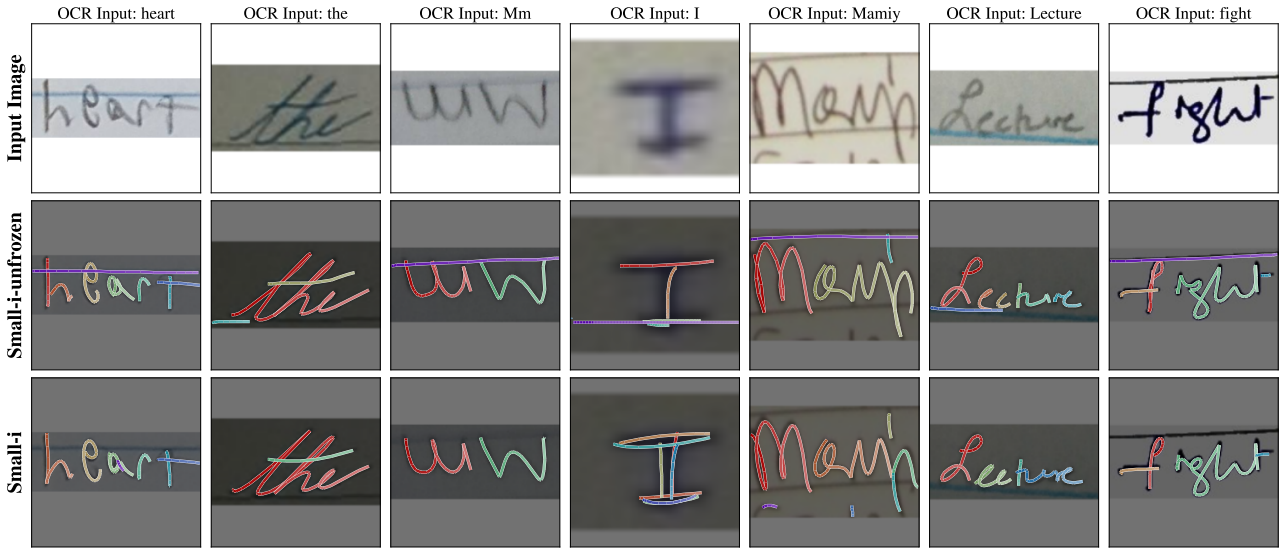

Figure 36: Difference between **Small-i-unfrozen** which was trained with an unfrozen ViT (seed 46 in Figure 35) and **Small-i** which was trained with a frozen ViT. This shows that **Small-i-unfrozen** is more sensitive to background noise compared to **Small-i**.

D.2. Inference type matters

As we first illustrated in Section 4.6, different inference tasks can produce different output digital inks especially when the input image of text is semantically or geometrically ambiguous due to either image quality or how the texts were written. For example the first column of Figure 37 where the word "wich" is removed by the writer with a strike through line but still recognized by both *Recognize and Derender* (model intrinsic) and *Derender with Text* (extrinsic OCR system) inference,

and the difference in understanding and locating the textual information in input image results in different output digital inks. Additionally, the output from *Vanilla Derender* where semantics are not intentionally involved during training is more robust to ambiguous inputs but shows poorer semantic consistency with the input image.

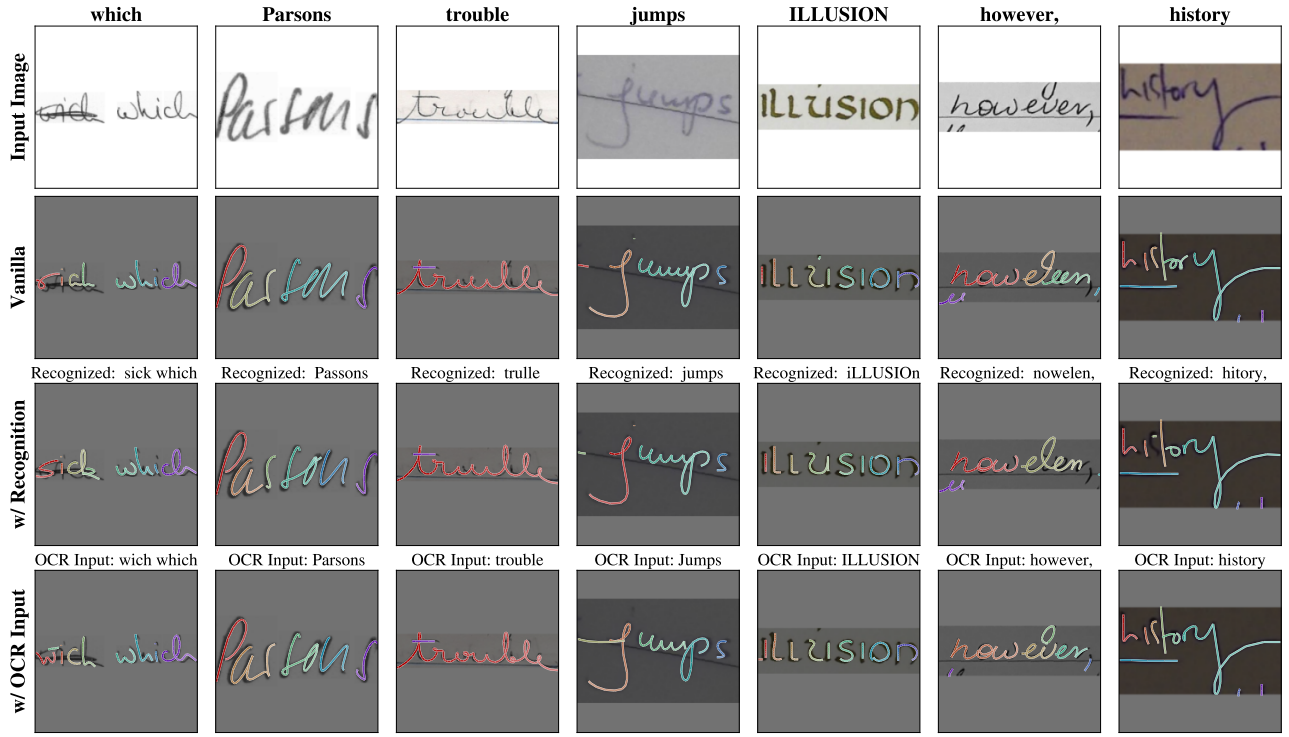

Figure 37: **Difference between inference tasks** of **Small-i** on samples from 3 public evaluation datasets, where the texts on top are the ground truth labels for the images in these datasets. **Vanilla** stands for inference with *Vanilla Derendering*, **w/Recognition** stands for inference with *Recognize and Derender*, and **w/ OCR Input** stands for inference with *Derender with Text* where we resort to an OCR system.

For *Derender with Text* inference, we have shown that how external textual input could benefit the model's semantic understanding. We explore the behavior of the model in cases where the external text is different than the ground-truth textual information in Figure 38. The results reveal that the model's output is not solely determined by input text. It demonstrates the ability to produce valid outputs even when external textual information is incomplete or erroneous.

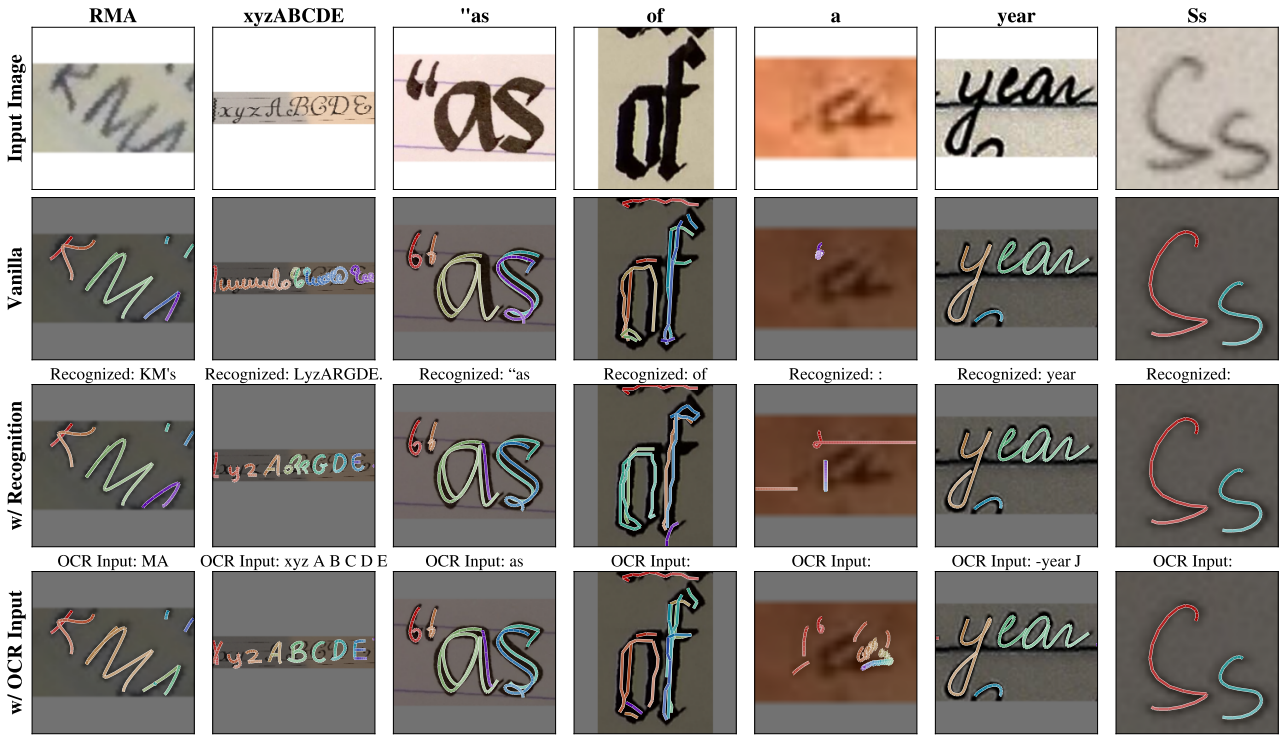

Figure 38: **Cases when there is a mismatch between ground-truth textual information and external textual information provided to the model.** We evaluated this using **Small-i** on samples from 3 public evaluation datasets, where the texts on top are the ground truth labels for the images in these datasets. **Vanilla** stands for inference with *Vanilla Derender*, **w/Recognition** stands for inference with *Recognize and Derender*, and **w/ OCR Input** stands for inference with *Derender with Text* where we resort to an OCR system.

E. Rendering and Data Augmentation

To create synthetic rendered inks of our online samples, we use the Cairo graphics library to render online ink samples onto a 224×224 canvas. We then add several augmentations to these rendered samples, to ensure that they are closer in domain to the real-world samples expected.

Before rendering, we first randomly rotate the samples. Then, we pick the color of the strokes and the background uniformly at random from the RGB color space and render the ink with a random width. Furthermore, we add lines, grids, or Gaussian noise into the background with a fixed probability. Finally, we potentially add box blur on the resulting image. This approach was mainly inspired by approaches from synthetic OCR generation, e.g. as described in (Etter et al., 2019). The detailed parameters of data augmentation and some samples are shown in Table 7. An example of how samples augmented with all parameters chosen at random could appear in the training set can be seen in Figure 39. Other approaches, such

Table 7: Data Augmentations Overview (sample from Aksan et al. 2018).

Augmentation	**Possible Values**	**Examples**
Rotation	angle (rad): $[-\frac{\pi}{4}, \frac{\pi}{4}]$	
Stroke color	RGB: $[0, 1]^3$	
Background color	RGB: $[0, 1]^3$	
Stroke Width	width: [1px, 12px]	
Lines	line width: [1px, 6px] line dist: [10px, 100px] line color: RGB: $[0, 1]^3$	
Grids	line width: [1px, 6px] line dist: [10px, 100px] line color: RGB: $[0, 1]^3$	
Gaussian Noise	standard dev.: [50, 500]	
Box blur	radius: [0px, 5px]	

as perspective skew, shifting, scaling, or padding, did not show any improvements in the model performance and were

discarded.

Figure 39: Demonstration of possible ways to combine all used augmentations. (Sample from Aksan et al. 2018)

F. Additional Model Details

The overview and individual components of three versions of our model **Small-i**, **Small-p**, and **Large-i** is shown in Table 8.

Table 8: Model overview and components.

Model	Components	Parameters	Training Data
Large-i	ViT-L mT5-Large	303M 783M	In-house proprietary datasets as shown in Appendix B.
Small-i	ViT-B mT5-Base	85M 247M	In-house proprietary datasets as shown in Appendix B.
Small-p	ViT-B mT5-Base	85M 247M	Publicly available datasets as shown in Appendix B.

Implementation Details. Similar to PaLI models (Chen et al., 2022a; 2023a;b), our models together with the training mixtures are implemented in `JAX/Flax` (Bradbury et al., 2018) using the open-source `T5X`, `SeqIO` (Roberts et al., 2023) and `Flaxformer` (Heek et al., 2023) frameworks. For training, we use the Adafactor optimizer (Shazeer & Stern, 2018) with $\beta_1 = 0$, second order moment of 0.8, and a language-model–style teacher forcing with softmax cross-entropy loss. For the learning rate schedule, we use the linear decay scheduler with a peak learning rate of 0.01, 5k steps of warmup and a linear decay with decay factor of $3e-6$, and a dropout of 0.1 on both ViT encoder and the mT5 encoder-decoder. We train our models for 340k steps with batch size 512. With frozen ViT encoders, the training of **Small-i** takes $\sim$33h on 64 TPU v5e chips and the training of **Large-i** takes $\sim$105h on 64 TPU v5e chips.

In our experiments, we observe marginal performance improvements in derendering tasks through adjustments in temperature sampling. However, for the sake of reproducibility and framework simplicity, we employ a greedy decoding strategy during inference for all tasks, with model checkpoints selected from the final step.

Training Mixture. As described in Section 3.2, the training mixture consists of 5 tasks constructed using SeqIO (Roberts et al., 2023). These tasks are pre-shuffled before training, ensuring each task appears with equal probability throughout the entire training process. And to foster reproducibility, the input pipeline as well as the model initialization are designed to be deterministic, reducing randomness-induced variations that could affect our design choices.

G. General Virtual Sketching Framework Performance

We demonstrate the performance of three GVS models from the official GitHub repository: Line Drawing Vectorization, Rough Sketch Simplification, and Photo to Line Drawing Conversion in two setups (original image as input and binarized image as input) in Figure 40. We inspect the samples visually on the more challenging HierText dataset. The photo to line drawing model was trained on people portrait photos and fails to generalize to the images of text. Binarization is beneficial for both remaining setups and reduces the noise when derendering texts. Line Drawing Vectorization and Sketch Simplification perform similarly on the inspected samples, we choose Line Drawing Vectorization on binarized images as a baseline since this model was trained on black & white raster images and therefore treats binarized samples as in-domain. For derendering of the sketches (Figure 6) we use the Sketch Simplification model as intended by the authors.

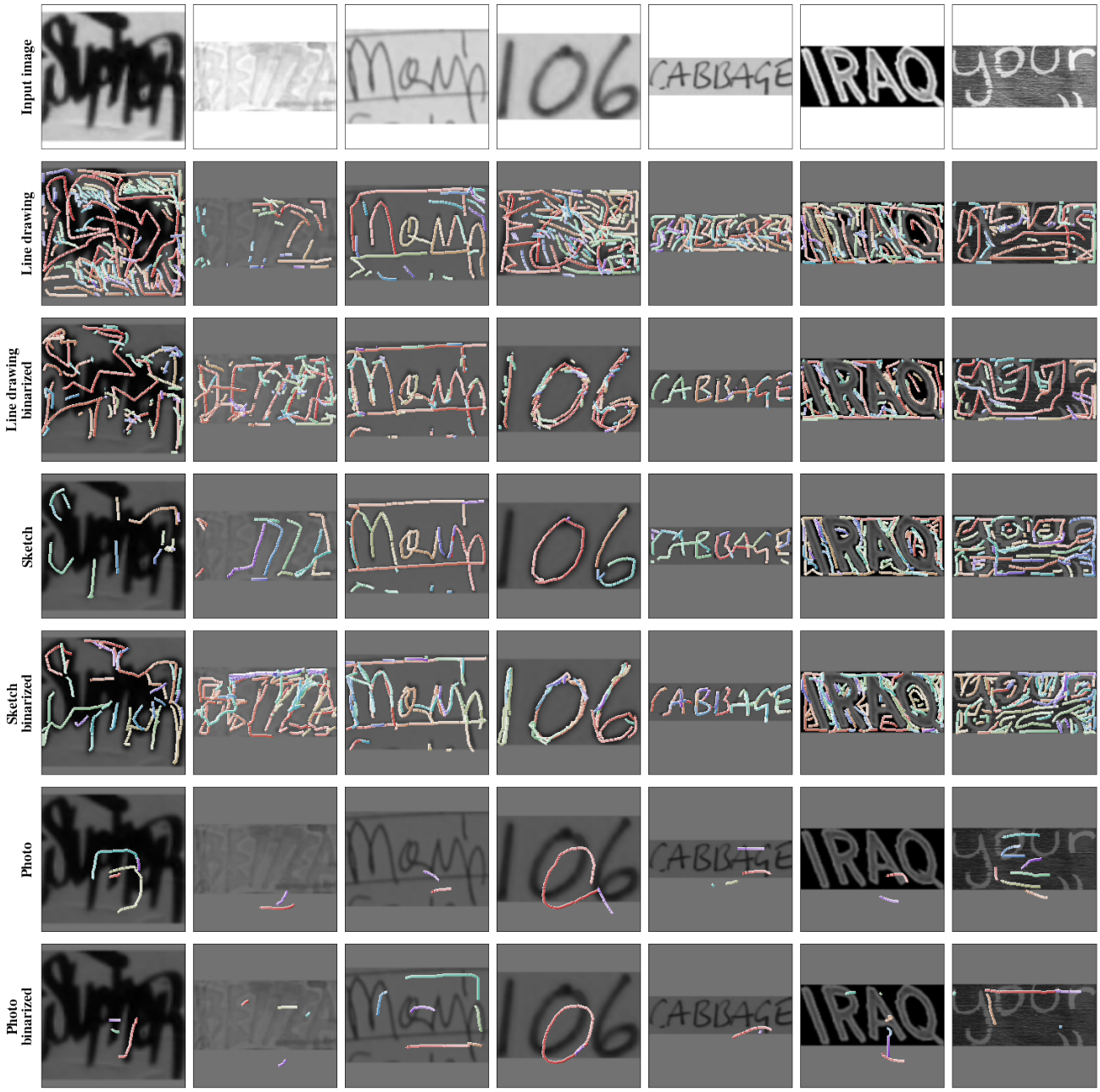

Figure 40: Comparison between General Virtual Sketching Framework inference setups.

H. Online Handwriting Recognizer Details

In this section we provide the result of online handwriting recognizers used to evaluate the low-level feature similarity of real inks for IAMOnDB dataset and inks derendered by **Small-p**, **Small-i** and **Large-i** (see Table 9). We observe that the inks acquired with all of our models perform similarly when used as training data both on their own and in combination with real digital inks.

Table 9: Online handwriting recognition results on IAMOnDB test set for models trained on real digital inks (IAMOnDB train set), derendered digital inks (from IAM train set) and the combination of the two.

	IAMOnDB	**Small-i IAM derendered**	**Small-i IAMOnDB + IAM derendered**	**Small-p IAM derendered**	**Small-p IAMOnDB + IAM derendered**	**Large-i IAM derendered**	**Large-i IAMOnDB + IAM derendered**
CER	6.1	7.8	4.6	8.2	**4.5**	8.4	4.6

Additionally, we describe the architecture and our recognition training procedure. We choose a common approach of training a (relatively small, 9.2M, due to the amount of the training data) Transformer encoder combined with a CTC loss, similar to Alwajih et al. 2022. We fix the same preprocessing steps for both real and derendered inks. Those include shifting and scaling the inks to start at the origin and have a fixed y side, time resampling at 20 ms for real inks and time hallucination (assuming the predicted ink tokens are produced at a fixed sampling rate of 20ms, matching the training data) and adding pen up strokes. We apply a random rotation within $\pm 45^\circ$ with probability 0.5 as data augmentation to account for the small size of the dataset. The points from the ink are encoded into Bezier curves. The input features are then processed by 7 transformer attention layers with 8 attention heads.

We train the models on a training set of IAMOnDB only (baseline), derendered IAM train set only and the combination of the two for 17.5k steps with batch size of 64. We perform a hyper-parameter search for dropout and learning rate on the same grid for all 3 setups. The best parameters for each model can be found in Table 10.

Table 10: Recognizer parameters found through hyper-parameter search.

Parameter	**IAMOnDB**	**Small-i IAM derendered**	**Small-i IAMOnDB + IAM derendered**	**Small-p IAM derendered**	**Small-p IAMOnDB + IAM derendered**	**Large-i IAM derendered**	**Large-i IAMOnDB + IAM derendered**
Dropout	0.25	0.25	0.3	0.3	0.25	0.25	0.25
Learning rate	0.005	0.001	0.001	0.005	0.005	0.001	0.001

I. Limitations

While our system exhibits satisfactory performance across various tested inputs, we have identified specific failure patterns that warrant further attention. As some examples shown in the Figures 36 and 38 we notice model's performance deteriorates when processing samples with wider stroke widths or significant variations in stroke width. Additionally, we observed that the model's ability to preserve handwriting details improves with the scaling of the vision component (visualized in Figure 5, aligned with findings in Section 4.5.1). However, we also noticed a lack of proportional scaling between the vision (ViT) and text-ink (mT5) components negatively impacts semantic understanding.

Furthermore, our investigation, as detailed in Section 4.6, demonstrated that unfreezing the vision heads allows the model to capture more input image details. Nevertheless, this enhancement comes at the cost of training instability and increased confusion between textual information and background noise. Future research in this domain may explore fine-tuning the vision backbone while leveraging the existing semantic understanding of ink representations, following a similar idea as described in (Zhai et al., 2022).

J. Human Evaluation

We provide the text of our evaluation instructions in Table 11 and the screenshot of the interface in Figure 41. Furthermore, we show the samples and corresponding rating (after majority voting) in Figure 42.

Table 11: Human evaluation campaign instructions.

Description	Instruction
Task setup	You will be looking at input images of photos of handwriting in a variety of styles and the corresponding digital ink tracing performed either by a human or by one of our derendering models. Each stroke is rendered in a different color and with color gradient (darker → whiter) to reflect the direction. You will see the derendered ink as well as the derendered ink overlaid on top of the image and can use either to make the judgment. For each sample answer the two questions on the right and click submit to proceed to the next sample.
Is Reasonable	Is the digital ink a reasonable tracing of the input image? • Yes, it's a good tracing • It's an okay tracing, but has some small errors • It's a bad tracing, has some major artifacts One interpretation of the quality of the ink could be like this: If you were to use it in a note taking app, would you • keep it as is: good tracing • need to make some edits: okay tracing • not use it at all: bad tracing If you need some examples to get a better feeling, you can use the ones below, however there is some ambiguity within the task since we don't want to impose any strict rules but allow for a natural variety of what people find a good derendering (it is designed to be subjective).
Is Real	Could this digital ink have been produced by a human? • Yes • No In this question go with your intuition, do you think the ink could have been traced by a [reasonable] human?

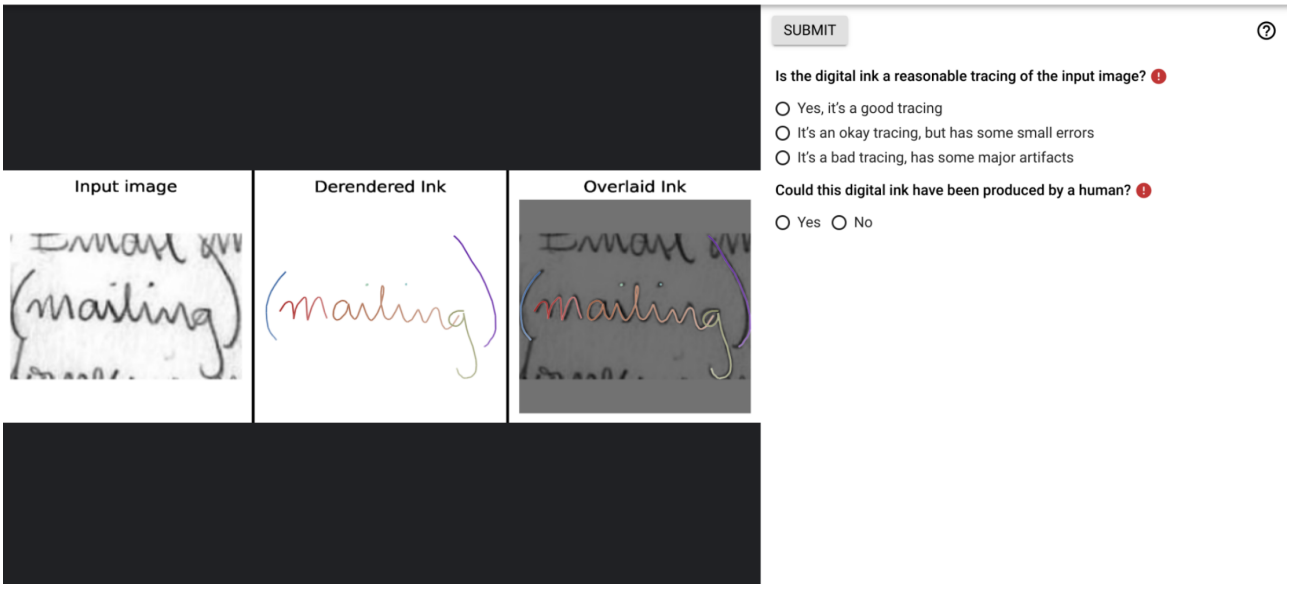

Figure 41: A screenshot of the human evaluation interface.

Figure 42: Examples of samples produced by our models and ratings given to them by our evaluators.

K. Model Card

We present the model card (Mitchell et al., 2019) of our public release model **Small-p** in Table 12.

Table 12: Model card of publicly available **Small-p**.

Model Summary	
Model Architecture	A multimodal sequence-to-sequence Transformer (Vaswani et al., 2017) model with the mT5 (Xue et al., 2021) encoder-decoder architecture. It takes text tokens and ViT (Dosovitskiy et al., 2021) dense image embeddings as inputs to an encoder and autoregressively predicts discrete text and ink tokens with a decoder.
Input(s)	A pair of image and text.
Output(s)	Generated digital ink.
Usage	
Application	The model is for research prototype, and the public version is planned to be released and available for the public.
Known Caveats	None.
System Type	
System Description	This is a standalone model.
Upstream Dependencies	None.
Downstream Dependencies	None.
Implementation Frameworks	
Hardware & Software	Hardware: TPU v5e. Software: T5X (Roberts et al., 2023), JAX/Flax (Bradbury et al., 2018), Flaxformer (Heek et al., 2023) For implementation details please refer to Appendix F.
Compute Requirements	Please refer to Appendix F.
Data Overview	
Training Datasets	The ViT encoder (Dosovitskiy et al., 2021) is pretrained on ImageNet-21k (Deng et al., 2009), mT5 encoder and decoder are initialized from scratch. The entire model is trained on the mixture of publicly available datasets described in Appendix B.
Evaluation Results	
Evaluation Methods	Human evaluation (reported in Section 4.5.1) and automated evaluations (reported in Section 4.5.2).
Model Usage & Limitations	
Sensitive Use	The model is capable of converting images to digital inks. This model should not be used for any of the privacy intruding use cases, e.g. forging handwritings.
Known Limitations	Reported in Appendix I.
Ethical Considerations & Potential Societal Consequences	Reported in Sections 6.1 and 6.2.

L. Additional Visualizations

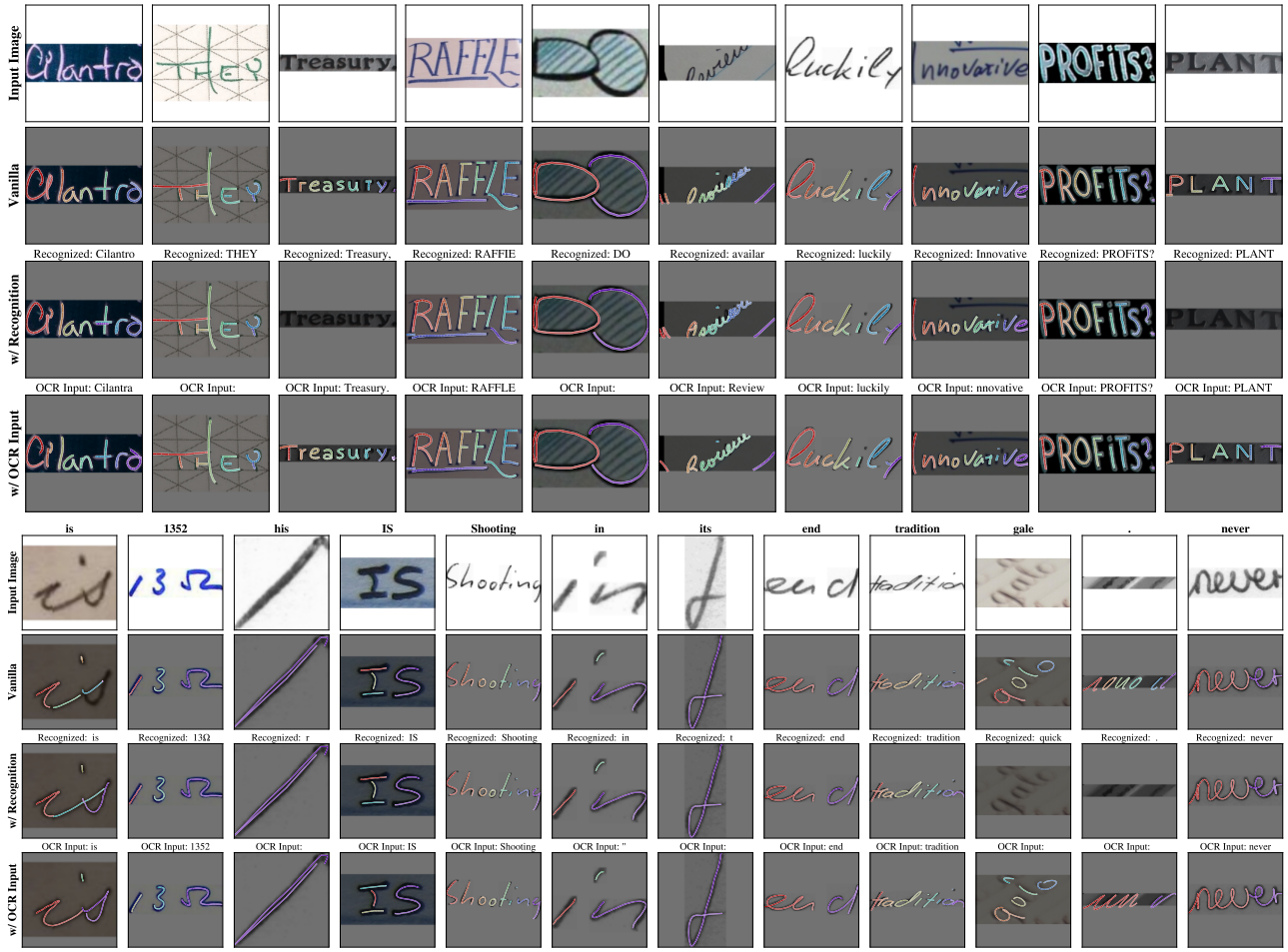

Figure 43: Results of **Small-p** on randomly selected samples from 3 public evaluation datasets.

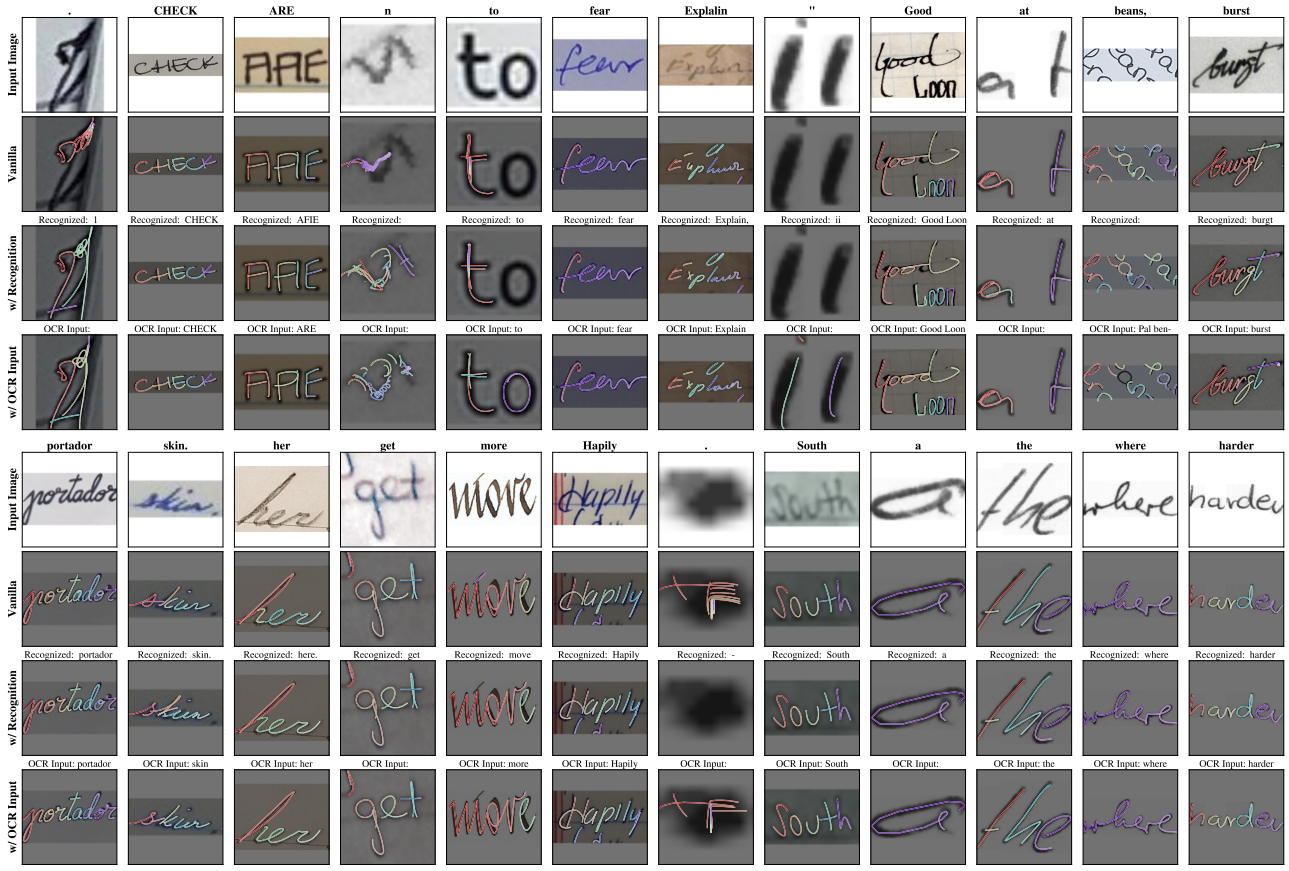

Figure 44: Results of **Small-i** on randomly selected samples from 3 public evaluation datasets.

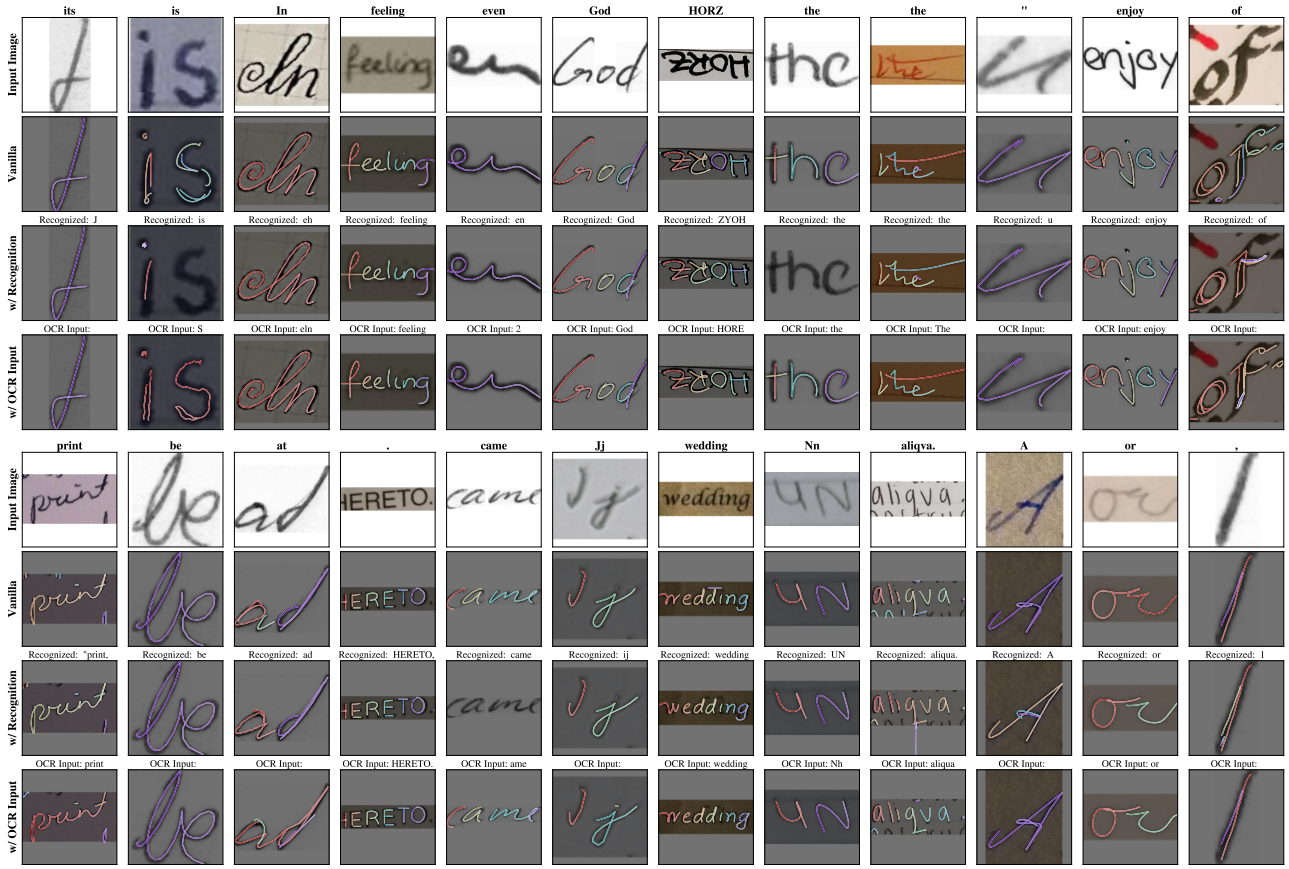

Figure 45: Results of **Large-i** on randomly selected samples from 3 public evaluation datasets.

L.1. More Visualizations on Public Evaluation Datasets

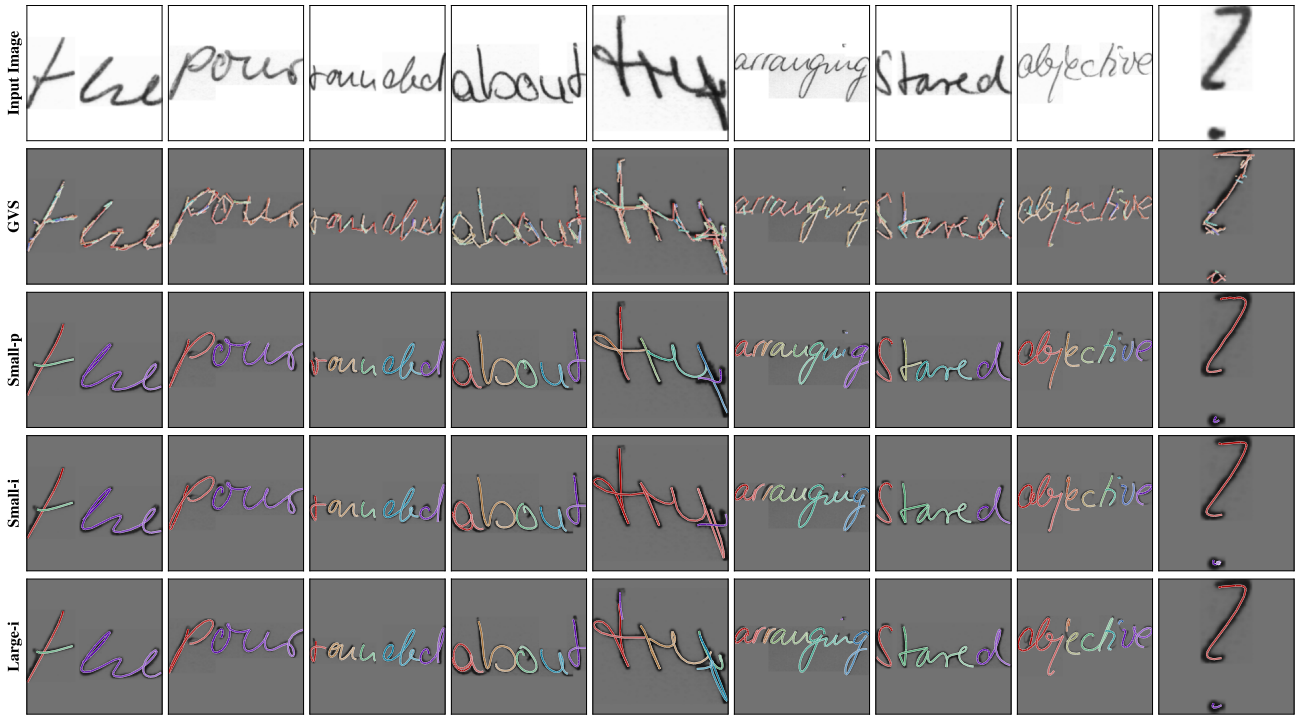

Figure 46: Comparison between **Small-i**, **Small-p**, and **Large-i** on IAM Dataset.

Figure 47: Comparison between **Small-i**, **Small-p**, and **Large-i** on IMGUR5K Dataset.

Figure 48: Comparison between **Small-i**, **Small-p**, and **Large-i** on HierText Dataset.

L.2. Open-ended Generation

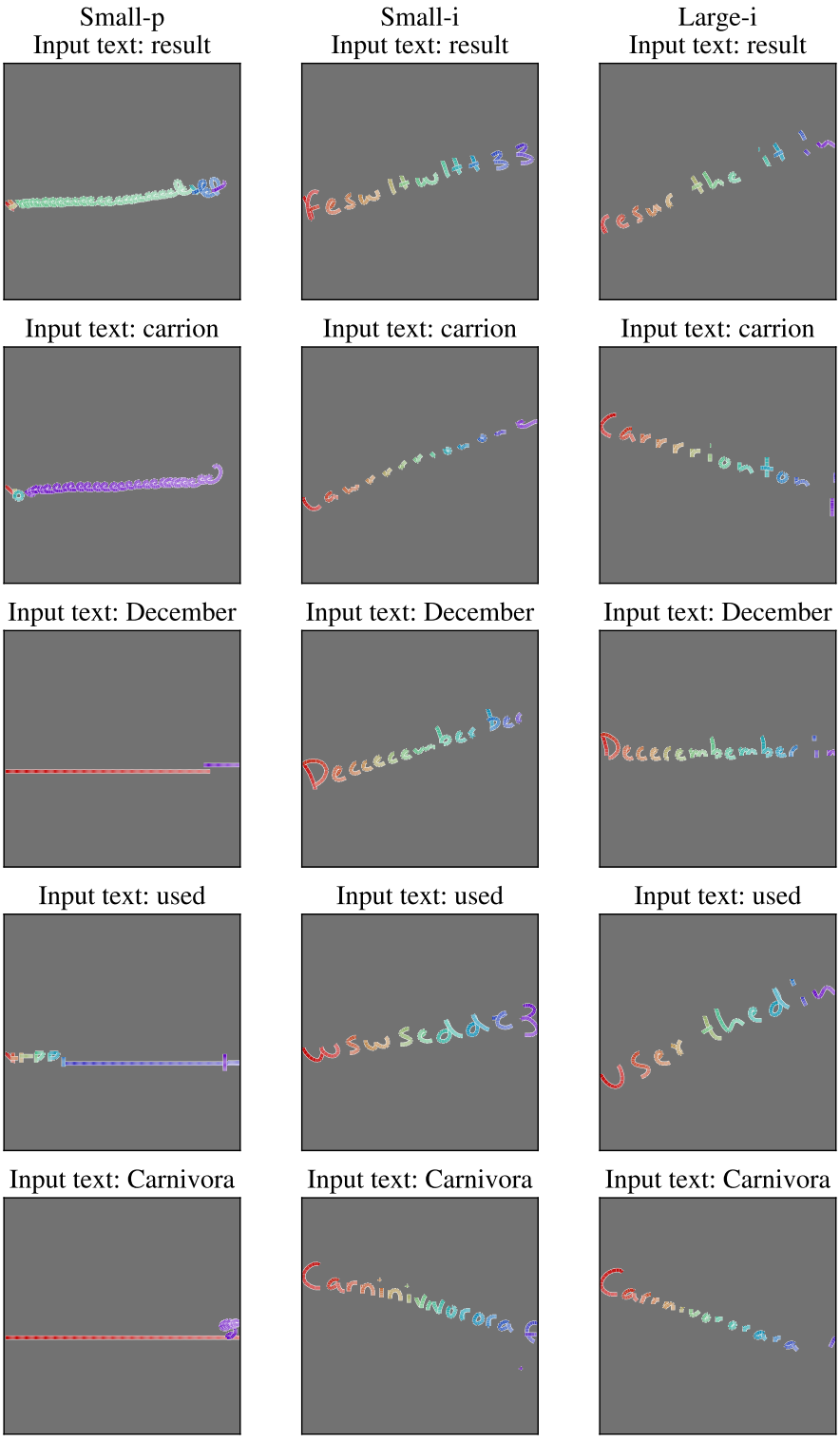

Figure 49: **Inability to perform open-ended generation.** Prompting **Small-p**, **Small-i** and **Large-i** with text that is not presented in the image does not produce digital inks that match the textual input.