SET, SORT! A Novel Sub-Stroke Level Transformers for Offline Handwriting to Online Conversion

Elmokhtar Mohamed Moussa, Thibault Lelore, Harold Mouchère

SET, SORT! A Novel Sub-Stroke Level Transformers for Offline Handwriting to Online Conversion

Elmokhtar Mohamed Moussa[1,2][0009−0009−7076−6400], Thibault Lelore[1][0000−0002−7083−2422], and Harold Mouchère[2][0000−0001−6220−7216]

[1] MyScript SAS, Nantes, France
{elmokhtar.mohamed.moussa, thibault.lelore}@myscript.com
[2] Nantes Université, École Centrale Nantes, CNRS, LS2N, UMR 6004, F-44000 Nantes, France
{elmokhtar.mohamedmoussa, harold.mouchere}@univ-nantes.fr

Abstract. We present a novel sub-stroke level transformer approach to convert offline images of handwriting to online. We start by extracting sub-strokes from the offline images by inferring a skeleton with a CNN and applying a basic cutting algorithm. We introduce sub-stroke embeddings by encoding the sub-stroke point sequence with a **S**ub-stroke **E**ncoding **T**ransformer (SET). The embeddings are then fed to the **S**ub-strokes **OR**dering **T**ransformer (SORT) which predicts the discrete sub-strokes ordering and the pen state. By constraining the Transformer input and output to the inferred sub-strokes, the recovered online is highly precise. We evaluate our method on Latin words from the IRONOFF dataset and on maths expressions from CROHME dataset. We measure the performance with two criteria: fidelity with Dynamic Time Warping (DTW) and semantic coherence using recognition rate. Our method outperforms the state-of-the-art in both datasets, achieving a word recognition rate of 81.06% and a 2.41 DTW on IRONOFF and an expression recognition rate of 62.00% and a DTW of 13.93 on CROHME 2019. This work constitutes an important milestone toward full offline document conversion to online.

Keywords: offline handwriting · transformer · online recovery.

1 Introduction

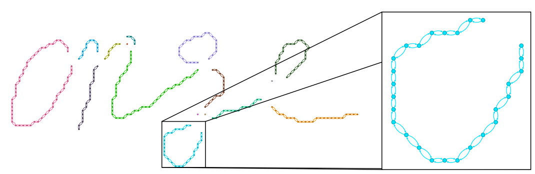
In today's highly virtual and automated world, note-taking is still a manual procedure. It is a ground to express our volatile thoughts and ideas, allowing their organization and the emergence of our creativity afterward. While pen and paper still offer unmatched comfort and efficient input methods for handwritten notes, it disables their exploitation to their full potential. They are usually digitized as offline documents by capturing images with a scanner or camera. This is an inconvenient step for most users and it also adds noise that affects offline processing systems. Online documents - the offline counterpart - are recorded on touch-sensitive surface devices with an e-pen, enabling a more powerful machine-automated organization and edition of handwritten documents, with intuitive pen gestures. Many commercial software specializing in note-taking exists, proposing a plethora of functionalities such as recognition, note indexing, collaborative note-taking, *etc.* The online domain is ever-evolving as the offline is already far behind. By developing an offline-to-online conversion system we allow the users to take to their advantage the best of the two modalities: ergonomic note-taking with a pen and paper and powerful editing and processing of the digital ink. Recently, attractive hybrid devices are surfacing. Their hardware closely mimics a pencil offering a more ergonomic input method while still proposing online processing tools. However, in the quest for paper-like hardware, the devices are still today limited in computational resources compared to other touch devices. Research efforts [17] have been conducted in the document analysis domain to automatically recover online from offline documents by retrieving the pen trajectory. Thus allowing for the direct exploitation of paper and pen notes in the existing online processing systems.

However, as datasets coupling online with offline are scarce [20,18], the applications of data-driven approaches remain limited. To overcome this issue, rasterization or online data to offline conversion is commonly used for training multi-modal systems. Converting online signals to realistic raster images often involves adding noise and simulating pen tip width and movement speed [7]. Other advanced applications use generative adversarial networks [11] to generate artificial papyrus and other historic documents. Multi-modal systems utilize both online and offline, combining temporality with spatial clues for better performances. For instance, handwriting recognition is typically classified into two types: offline and online systems. Multi-modal Handwriting recognition systems [21,22] are shown to outperform their mono-modal counterparts. In this paper, we focus on the reverse problem, which is offline to online conversion. Vectorization similarly tries to model a line drawing image as a set of geometric primitives (polygons, parametric curves, *etc.*) corresponding to elements found in Scalable Vector Graphic (SVG) format. It is mainly applied to technical drawing [6] and 2D animation sketching [7]. In this particular application, retrieving temporal ordering between the extracted vector elements is less relevant.

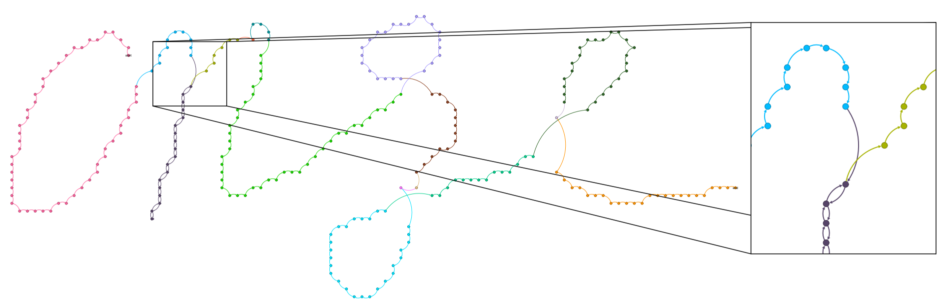
For handwriting applications, we are more involved in the recovery of pen trajectory from images. The availability of temporal information in online systems often makes them better performing than their offline analog [16]. In 2019, the Competition on Recognition of Online Handwritten Mathematical Expressions (CROHME) [12] included for the first time an offline recognition task. It has since sparked a great interest in offline to online conversion [4] in this specific domain. Classical approaches are rule-based systems. They usually operate on a topology to detect regions where the drawing direction is ambiguous (*e.g.* junctions) and employ a set of handcrafted heuristics to simplify and resolve them. However, they are very hard to maintain and do not generalize to different languages or content. Recently, many data-driven approaches have been proposed in the literature to recover online from offline. However, CRNNs models [3,2] rely on fixed-size feature maps of the whole offline image, regardless of the ink density, to predict all the underlying intricacies in the temporality of the different strokes resulting in the partial or full omission of strokes. In this paper, we propose the following contributions:

- We propose novel sub-stroke level Transformers (SET and SORT) to recover the online from offline (see fig.1) instead of CRNNs architectures [3,2].
- We move from the image to sequence framework to operate on the sub-stroke level to perform a local and global analysis of the different junctions as is adopted in classical approaches [4].
- Our SET and SORT approach outperforms prior online recovery work on the handwritten text of the IRONOFF dataset. We also extend our work to more complex maths equations of the CROHME dataset.

(a) Input offline image.



(b) Extracted sub-strokes from the CNN inferred skeleton. A sub-stroke can be drawn in both directions.



(c) Network online prediction. *start* and *end* nodes are annotated. The zoom box shows the predicted sub-stroke direction and ordering as illustrated by arrows. The edge color is the destination node. For double-traced sub-strokes, edges are read in the clockwise orientation as in [9].

Fig. 1: Given an input offline image (a), sub-strokes incident on the same junctions are extracted (b). Our Network predicts the sub-stroke's order and directions (c). The trivial longest path obtained by following the outgoing edges from *start* node to *end* node is the predicted online signal.

2 RELATED WORKS

Line drawing vectorization is a crucial step in the creation of 2D animations and sketches. It involves converting drawing images into vector graphics. Artists often begin by sketching their work on paper and then manually vectorizing it digitally for finalization. However, vectorizing rough and complex real-world sketches can be challenging, as multiple overlapping lines need to be combined into a single line and extraneous lines and background noise need to be removed. [19] proposed a fully convolutional simplification network with a discriminator network to clean high-resolution sketches. [7] developed a two-step system employing two neural networks to vectorize clean line drawings. The first multi-task CNN predicts skeleton and junction images, and the second CNN resolves the segment connectivity around the junctions. They demonstrate state-of-the-art results on the public *Quick, Draw!* [8] dataset. However, their method is restricted to relatively small junctions of degrees 3 to 6 that fit in a 32×32 window.

Pen trajectory recovery Throughout the years, numerous methods have been proposed by researchers to tackle the task of pen trajectory recovery from offline images. The steps involved in these methods typically include extraction of topology and detection of local ambiguous regions such as junctions and double-traced strokes. These ambiguities are then resolved using hand-designed rules. The existing methods can be broadly categorized into three types: recognition-based, topology-based, and tracking-based. Recognition-based [5] methods, which were first introduced for drawings composed of regular shapes such as diagrams and engineering drawings, detect these shapes by fitting geometric primitives. This approach is not ideal for handwritten text due to limitations in the possible graphical representation. Topology-based methods [10] construct a representation using topological information from the image (skeleton, contour, *etc.*) and view pen trajectory recovery as a global or local optimization problem. [17] developed a weighted graph approach that finds the best matching paths for pen trajectory recovery and demonstrated good performance on English characters. The tracking-based approach estimates the pen's relative direction iteratively. [24] proposed an image-to-sequence framework to generate pen trajectories using a CNN and fully connected layers without any RNN. This approach showed good results on Chinese and English handwriting datasets but the model's complexity is directly proportional to the image resolution. Moreover, their method requires a skeleton as input and inferred skeletons can be noisy and very different from the perfect skeletons their network was trained on, leading to unexpected failures at test time. [14] investigated the generalization of the previous approach to arbitrary-size images of math equations. They suggest using a fully convolutional neural network trained on noisy offline images. The network learns to predict both a skeleton and the next pen positions. However, the lack of temporal modeling causes over-segmentation of long strokes. Other lines of research followed a sequence modeling approach with CRNNs. In [3] a CNN-BLSTM network was proposed. They obtain good results on Tamil, Telugu, and Devanagari characters. However, this approach is limited to single isolated characters and requires separate models for each script. [2] extended the same CRNNs network to the text line scale [13] by applying a variety of data-augmentation techniques and an adaptive ground-truth loss to counter pathological strokes impact on the model learning. Their system is shown to recover a great portion of the online signal but still tends to omit some small strokes or even to over-simplify complex long strokes. Moreover, this approach is not well suited for 2D content such as math equations. In fact, resizing larger images of equations to a small fixed height (61 pixels) can lead to illegible content.

Stroke embeddings and Transformers Most of the proposed aforementioned approaches rely on the sequence-based networks to recognize drawing [8] or handwritten text [3]. [1] propose stroke-level Transformers to embed strokes into fixed lengths representations that are used to generate auto-completion of diagrams drawings. They show that Transformers outperform the sequential RNN approach [8]. However, they conclude that cursive handwriting strokes are challenging and longer strokes can't be correctly encoded in a fixed-size embedding. In this work, we model sub-stroke as embedding. Sub-strokes are much simpler shapes (straight lines, short open curves, *etc.*) that are far easier to model.

3 SET, SORT: Sub-Stroke Level Transformers

After an overview of the proposed system, we present the sub-stroke extraction algorithm, the **S**ub-stroke **E**ncoding **T**ransformer (SET) and the **S**ub-strokes **OR**dering **T**ransformer (SORT).

3.1 Overview

We propose a novel sub-stroke ordering Transformer model to reconstruct the online signal from offline images. We start by using the FCNN from [14] to extract a skeleton (1-pixel thick outline) from the input offline image. A sub-stroke cutting algorithm based on junction detection is then applied to the extracted skeleton. We use a Transformer auto-encoder to learn sub-stroke embedding [1]. Finally, an auto-regressive Transformer decoder is used to predict the sub-strokes ordering using their embeddings. Fig. 2 shows an overview of our pipeline. More formally, given a set of sub-strokes $V = \{ss_1, ss_2, \ldots, ss_N\}$, with a sub-stroke defined as a sequence of coordinates $ss_i = (x_k, y_k)_{k=1}^{m}$. Each sub-stroke from the skeleton appears twice, in both directions. The goal is to predict the sequence indicating the writing order of the different sub-strokes $S = (o_1, o_i, \ldots, o_M)$ $o_i \in \{1, \ldots, N\}$ and how they should be merged to form strokes. This is achieved by predicting a pen-up to indicate the end of the stroke. We note that V and S can be of unequal lengths, for instance, sub-strokes can be ignored (as noise), used several times (redrawing), and most of the time sub-strokes are drawn in one direction only, therefore the opposite sub-stroke is omitted.

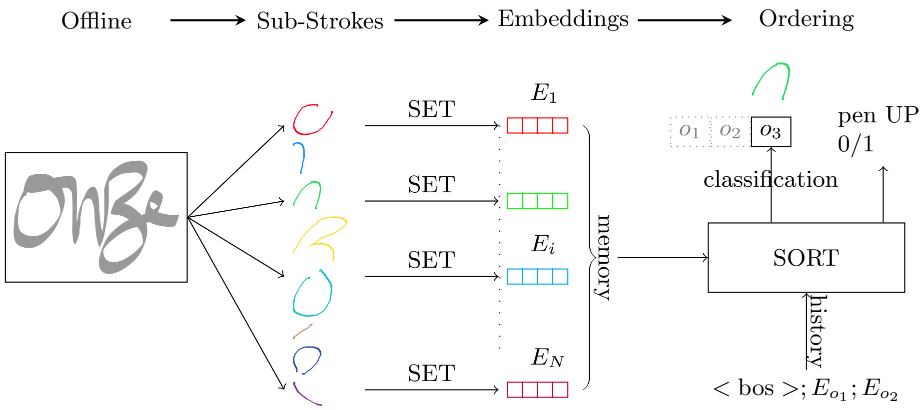


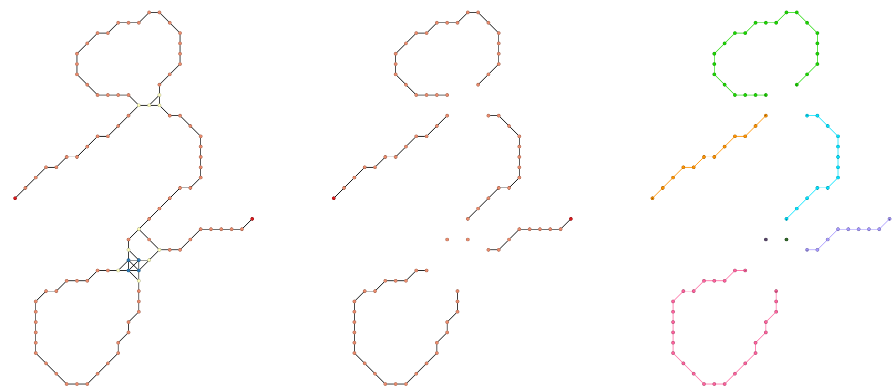
Fig. 2: Overview of our approach for the generation of the first three sub-strokes. After sub-stroke extraction, the SET encoder provides embedding for each sub-stroke. The SORT network uses the memory and the history to predict the next sub-stroke and the pen state. The sequence begins with a *bos* token and will end after M tokens with the eos token. Here we show the first three timesteps.

3.2 Sub-strokes extraction

After extracting a skeleton using an already trained FCNN [14], we apply a thinning algorithm [23] to remove the few small remaining ambiguities in the skeleton, obtaining I_{thin}. We cut the skeleton into sub-strokes by removing the different junctions pixels and computing the resulting connected components. A junction pixel is defined as a skeleton pixel with 3 or more 8-connected skeleton pixels. Each connected component will have two extremities, the skeleton pixels with exactly one 8-connected skeleton neighbor (see fig. 3). We compute the path from one extremity to another to define a sub-stroke. The opposite traversal path is also included as a distinct sub-stroke. Using this simple heuristic-free algorithm allows us to generalize to any handwritten content. Our sub-stroke cutting algorithm results in a normalization of stroke drawing. Partial inconspicuous stroke retracing is removed.

3.3 SET: Sub-stroke Embedding Transformer

We adapt the stroke embedding from [1] to the lower level of a sub-stroke. In fact, learning meaningful fixed-size vector representation for a stroke of arbitrary size and complexity can prove to be especially difficult for cursive handwriting. Sub-strokes are usually much simpler geometric primitives that are far easier to model. We define the sub-stroke auto-encoder as a Transformer followed by a sub-stroke reconstruction MLP, as shown in fig. 4. Before being embedded by the encoder, the input sub-stroke points are shifted to start at the origin and



(a) Skeleton pixels 8-connectivity, in yellow and blue junction pixels.

(b) Junctions pixels are removed, splitting the skeleton into segments.

(c) Connected components are labeled to obtain the sub-strokes. Extremities are marked by a cross.

Fig. 3: Illustration of the sub-strokes cutting algorithm.

normalized w.r.t. to the offline image dimensions. This ensures that embedding only captures important local geometric features.

Encoder Given an input sub-stroke defined as a sequence of points $ss_i = ((x_1, y_1), \dots, (x_m, y_m))$, the points are first linearly projected to vectors of size 64 and summed with a sinusoidal positional encoding of each timestep. The input embedding is then fed through a Transformer with a stack of 6 layers, and 4 attention heads, with a model dimension of 64 and a feed-forward size of 256. The decoder output vector for the last timestep n of ss_i is projected linearly to a vector of size 8 corresponding to the sub-stroke embedding E_i.

Decoder The sub-stroke reconstruction $F(E_i, t) \in \mathbb{R}^2, t \in [0, 1]$ is a parametric approximation of the sub-stroke curve using a two-layer MLP. It estimates the coordinates of the sub-stroke curve at every timestamp t. It's composed of a hidden layer of size 512 followed by *ReLU* and an output layer of size 2 corresponding to the coordinates (x_t, y_t) of a point. The auto-encoder stroke embedding network objective is to reconstruct accurately the input sub-stroke.

3.4 SORT: Sub-stroke Ordering Transformer

We present a novel sub-stroke ordering auto-regressive transformer based on the sub-stroke embedding. Each sub-stroke embedding is concatenated with the positional embedding of its starting point $[f(ss_i[0]); E_i]$ to add global information of the sub-stroke spatial arrangement in the offline image. We use a stack of

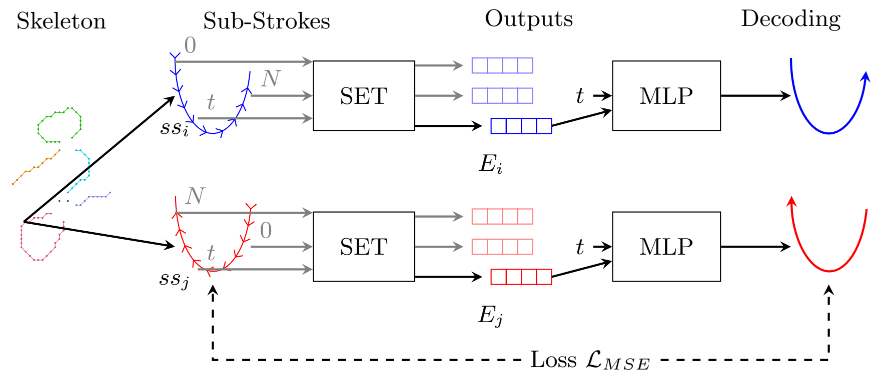


Fig. 4: Sub-stroke encoding transformer.

n_{layers} encoder-decoder Transformer with a model size of d_m and n_{heads} attention heads. The last transformer decoder layer employs a single attention head to compute the cross-attention between the encoder's output keys K and values V and the decoder self-attention output queries Q. This layer's output attention scores $\hat{A}_i$ over the sub-strokes set are used as predictions for the next sub-stroke ss_{i+1} probability distribution. As we can see in equation 1, the SORT outputs two decisions. On one hand, the attention scores over the sub-strokes set are used as predictions for the next sub-stroke ss_{i+1} probability distribution $\hat{A}_i$. On the other hand, the values O_i are used to predict the pen up state $\hat{P}_i$ with a small classification MLP.

$$\begin{aligned} \mathrm{Q}_i, \mathrm{K}_i, \mathrm{V}_i &= \mathrm{QW}_i^Q, \mathrm{KW}_i^K, \mathrm{VW}_i^V \\ \hat{A}_i &= \mathrm{softmax}\left(\frac{\mathrm{Q}_i \mathrm{K}_i^\top}{\sqrt{d_k}}\right) \in \mathbb{R}^{T \times L} \\ O_i &= \hat{A}_i V_i \\ \hat{P}_i &= \mathrm{MLP}\left(O_i\right) \\ \text{with } \ \mathrm{W}_i^Q &\in \mathbb{R}^{d_m \times d_k}, \mathrm{W}_i^K \in \mathbb{R}^{d_m \times d_k}, \mathrm{W}_i^V \in \mathbb{R}^{d_m \times d_v} \end{aligned} \tag{1}$$

To alleviate the lack of coverage we employ the Attention Refinement Module (ARM) [25].

3.5 Training

The SET network is trained separately from the SORT. We sample five points at random $t \in [0, 1]$ from the sub-stroke latent representation E_i by using $F(E_i, t)$. The network is trained with an *MSE* loss between the reconstructed points and the ground-truth sub-stroke points as in eq. 2.

$$\mathcal{L}_{MSE} = \frac{1}{5}\sum_{n=1}^{5}(F(E_{ss_i}, t_n) - ss_{i_{t_n}})^2 \tag{2}$$

We use teacher forcing to train the SORT network, it predicts the $i+1$ sub-stroke probability distribution with $\hat{A}_i$ and the associated pen state $\hat{P}_i$ given the i sub-stroke. The network is trained with a multi-task loss $\mathcal{L}$ combining a cross-entropy classification loss for sub-stroke ordering $\mathcal{L}_O$ and a binary cross entropy loss for pen state classification $\mathcal{L}_P$.

$$\begin{aligned} \mathcal{L} &= \lambda_1\mathcal{L}_O + \lambda_2\mathcal{L}_P, \\ \mathcal{L}_O(\hat{A}, A) &= \frac{1}{|V|}\sum_{y=0}^{|V|} -A_y \log\left(\hat{A}_y\right), \\ \mathcal{L}_P(\hat{P}, P) &= \frac{1}{2}\sum_{y\in\{\text{ down,up }\}} -\left(P_y \log\left(\hat{P}_y\right) + (1-P_y)\log\left(1-\hat{P}_y\right)\right), \end{aligned} \tag{3}$$

where $\lambda_1, \lambda_2 \in \mathbb{R}$ and P, A are respectively the ground-truth pen-state and sub-stroke successor. Sub-strokes are extracted from an accurate but still not perfect inferred skeleton. They are ordered using the ground-truth online signal to obtain A used to train our network. This ordering is defined as the oracle machine's solution to the sub-strokes ordering problem. The oracle ordering is obtained by using the original online to map each extracted sub-stroke from the offline image independently to a sub-section of the online. They are then ordered using their time of apparition in the online signal. We note that this oracle answer is a satisfying approximation of the original online. However, it can still introduce a small disparity from the original online, particularly in cases of invisible pen-ups or erroneous skeletons.

3.6 Inference

At inference time, we follow the same pipeline to extract sub-strokes from an offline image as explained in section 3.2. Sub-stroke embeddings are then produced using the SET network. The SORT network then iteratively predicts the next sub-stroke and corresponding pen state. We select the sub-stroke with the highest predicted probability as the next one which will be fed as input for the next timestep. Inference ends when a special *eos* token is predicted as the next sub-stroke. The result is a sequence of sub-strokes that we linearly interpolate to fill in the void left between two consecutive sub-stroke extremities (see fig.3), only when the pen state is "down" (i.e. $\hat{P}_i < 0.5$).

4 Evaluation

The goal of our method is to reconstruct accurately the pen trajectory reflected by a user's offline drawing. To quantitatively evaluate the quality of the online reconstructions, we employ two evaluation metrics DTW and handwriting recognition rate. While the DTW strictly measures geometric reconstruction fidelity, the recognition rate is a more lenient metric that measures semantic coherence.

4.1 DTW point-wise and DTW point-to-segment-wise

We compute a DTW distance between the inferred online signal and the ground truth signal to measure the accuracy of the network prediction. We also employ a modified DTW with a point-to-segment distance DTW_{seg} by [15] which is less sensitive to the sampling rate. We also evaluate the stroke extraction by using a DTW on the stroke level. Similar to the offline Stroke IoU proposed by [7], we use an online stroke DTW defined as :

$$\mathrm{SDTW} = \frac{1}{n} \sum_{i=1,\ldots,n} \min_{j=1,\ldots,m} \mathrm{DTW}(S_i, \hat{S}_j), \tag{4}$$

where S_i are ground-truth strokes and $\hat{S}_j$ are predicted strokes. This metric is useful to detect under/over-segmentation issues of strokes which are otherwise not taken into account by DTW.

4.2 Handwriting Recognition Rate

The natural variability in writing styles makes it so that different reconstructions are plausible. DTW-based metrics continuity constraint strictly matches two online signals, which leads to high-cost alignment in some cases such as delayed strokes, interchangeable strokes and reversed strokes. An online automatic handwriting recognition system can be used to recognize the retrieved online signal. The recognition results can be compared with a ground-truth text label, computing a word and character recognition rate (*WRR* and *CRR*) for handwritten text and expression recognition rate for handwritten math. This results in higher-level evaluation which is far less sensitive to writing styles. However, we note that powerful state-of-art recognition systems can correctly predict the text even if some symbols are approximated roughly. In our case, this is problematic since the predicted signal is no longer loyal to the user's handwriting. For this reason, it is important to supplement the recognition rate with the DTW to also account for visual accuracy. We use the MyScript interactive ink recognition engine version 2.0 [3] to evaluate the recognition.

5 Experiments

In this section, we present the training protocol and the evaluation results of our approach using online metrics.

[3] MyScript iink SDK is available at https://developer.myscript.com/docs/interactive-ink/2.0/overview/about/

5.1 Datasets and Training

Our networks are trained and evaluated on IRONOFF [20] and CROHME [12] datasets. We follow the same procedure as [14] to render synthetic offline images from their online counterpart. Our rendered offline images are noisier than the constant stroke width rendering proposed in [12], as we want to better mimic the end goal real word noisier offline images (cf. fig. 5). The training set of IRONOFF and CROHME contains respectively $48K$ and $10K$ samples, roughly equating to a total of $100K$ strokes each. We supplement CROHME with $15K$ equations from our private proprietary dataset.

(a) Random stroke width (b) Constant stroke width

Fig. 5: Comparison between variable and constant thickness stroking.

We first train the SET network on IRONOFF and freeze it during the training of the SORT network. The SORT network is trained on IRONOFF and fine-tuned on CROHME and our private datasets. We use the Adam optimizer with a learning rate of 0.001 and a batch size of 10. The training is performed on a single NVIDIA GeForce RTX 2080 Ti GPU with 24GB of memory and takes 20 hours to be completed.

5.2 Results

We evaluate and benchmark our method against state-of-art offline to online conversion systems. Table 1 shows the results on the test set of *IRONOFF* containing $17K$ test samples. Row (d) shows that the oracle approximation is very close to the original online (e). The small difference reflects the previously mentioned errors and simplifications. Our method (c) outperforms other state-of-the-art approaches (a) and (b) while using a relatively lighter model compared to the other data-driven approach of [2]. However, as shown by (d) there is still a margin for progression.

We also evaluate our method on the CROHME 2014 and 2019 test sets. The evaluation results of [4] and our method are presented in Table 2 and 3.

Our approach achieves a better stroke extraction resulting in higher expression recognition rates. As reported by rows (d) of tables 2 and 3 respectively, better online level DTW is not always synonymous with more precise stroke segmentation and more accurate recognition. In fact, [4] obtains a slightly better DTW of 14.29 as reported by table 3 however a fairly lesser stroke DTW 7.19 compared to ours of 3.85. The same applies to ExpRate as well, 57.01% compared to 62.00% of our approach.

Method	Parameters	DTW↓	DTW_{seg}↓	CRR↑	WRR↑
(a) CNN-BiLSTM [2]	7M	7.09	7.45	59.22	41.43
(b) Chungkwong et al. [4]	-	5.75	5.06	73.45	60.00
(c) Sub-stroke Transformer (Ours)	2M	**3.25**	**2.72**	**90.85**	**81.06**
(d) Oracle	-	0.33	0.32	92.56	83.45
(e) online GT	-			93.03	83.81

Table 1: Results on *IRONOFF* test set.

Method	DTW↓	DTW_{seg} ↓	SDTW↓	ExpRate↑
(a) Chungkwong et al.	16.30	16.13	6.54	52.43
(b) Sub-stroke Transformer (Ours)	24.54	24.37	12.29	29.37
(c) fine-tune (b) on CROHME	**13.75**	**13.59**	4.43	53.75
(d) fine-tune (c) on private datasets	13.93	13.80	**2.93**	**59.31**
(f) Oracle	0.24	0.22	0.50	66.63
(g) GT online	—	—	—	69.77

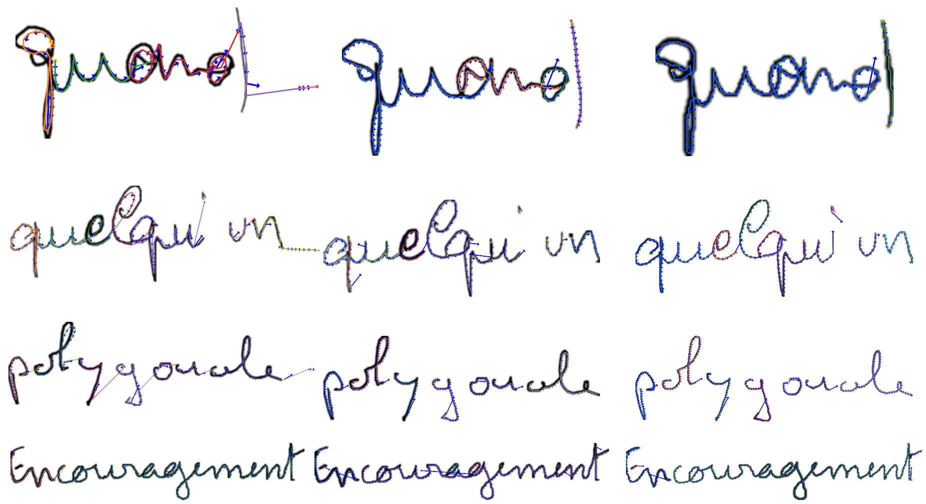
Table 2: Results on CROHME 2014 test set.

Method	DTW↓	DTW_{seg} ↓	SDTW↓	ExpRate↑
(a) Chungkwong et al.	**14.29**	**14.14**	7.19	57.01
(d) fine-tune on private dataset	14.98	14.80	**3.85**	**62.00**
(f) Oracle	0.26	0.24	0.69	70.19
(g) GT online	—	—	—	73.13

Table 3: Results on CROHME 2019 test set. Row (d) reports the results of the fine-tuned model on equations from CROHME and our private dataset.

Figure 6 shows a visual comparison between our approaches and other state-of-the-art methods on IRONOFF. Our approach 6c is observed to cover very closely most of the offline image compared to 6a and 6b. In fact, some characters are missing in their online reconstructions. For example, the smaller "e" loops (rows 3 and 4), the middle horizontal bar of "E" (row 4) and the apostrophe (row 2) are not covered. 6a and 6b tend to over-segment the strokes, on the other hand, our approach predicts more accurate pen ups resulting in a far less number of strokes. 6a often struggles with end-of-sequence predictions (first three rows).

Our method is observed to better capture the greater diversity in the stroke 2D ordering in Maths equations as illustrated by Figure 7. For instance, the reconstruction in 7b shows greater variability compared to the strict X-Y ordering of 7a. Here the superscripts are predicted after the exponents and the operators. This is a less common way to write but is still plausible. As highlighted in 7b



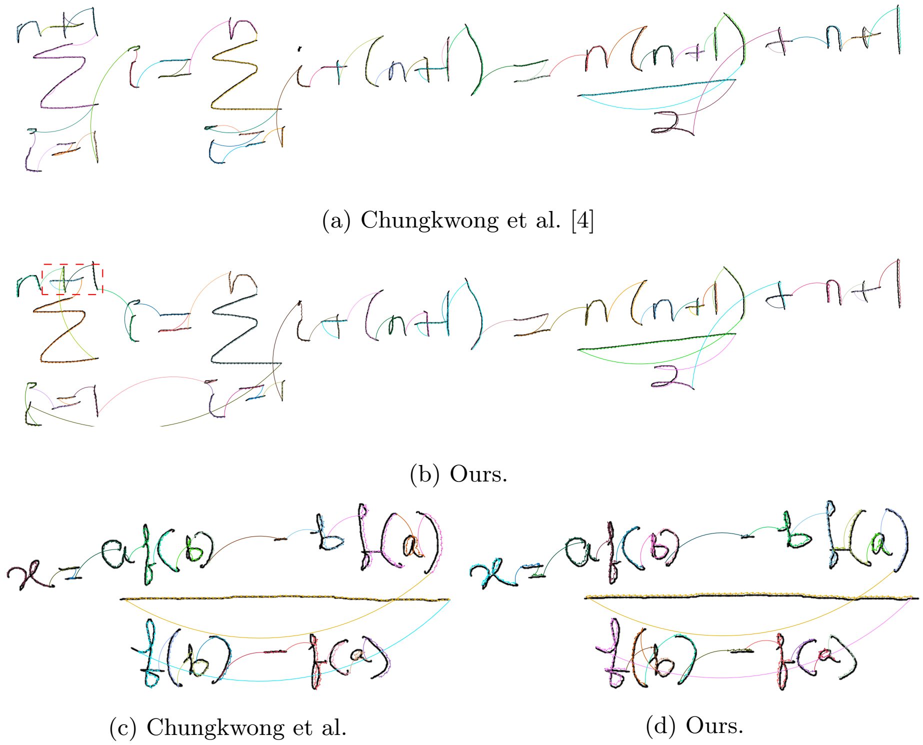
(a) Archibald et al. (b) Chungkwong et al. (c) Ours.

Fig. 6: Comparaison of our approach (c) to [2] (a) and [4](b) on IRONOFF samples. Each stroke is drawn with a distinct color. Blue arrows show the direction. The first and last stroke points are respectively yellow and red.

our network mistakenly re-crosses the first "+" sign (as highlighted in the red box), instead of drawing the "1", but is still able to recover the remaining strokes correctly. We hypothesize that it's due to the two strokes being of very similar shape and in close proximity to each other. We observe a few errors in 7a, the "i" dot is missing in one case. The ordering of the superscripts in the second "$\sum$" is far from ideal. In 7d, the parenthesis and their content are not always in the same order. Reflecting once again on the great diversity captured by our network. Thanks to our pen state prediction, we can more accurately segment the symbols 7d. In 7c the f in the term "$f(a)$" of the numerator is incorrectly segmented resulting in a bad recognition. Figure 8 shows the SORT prediction of the probability distribution of the next sub-stroke at every timestep. We observe that the network is very confident in its predictions and they are well centered around a small local region of the image. The network's reconstructed signal overall reflects the same temporal dynamics as the ground truth online signal. However, as depicted in 8b, in rare instances it locally drifts from the ground truth online.

6 Discussion

Our approach is able to generalize to different handwriting domains. By transferring the learned knowledge from Latin words to Maths equations we are able to achieve better results compared to handcrafted rule-based systems. However,



(a) Chungkwong et al. [4]

(b) Ours.

(c) Chungkwong et al.

(d) Ours.

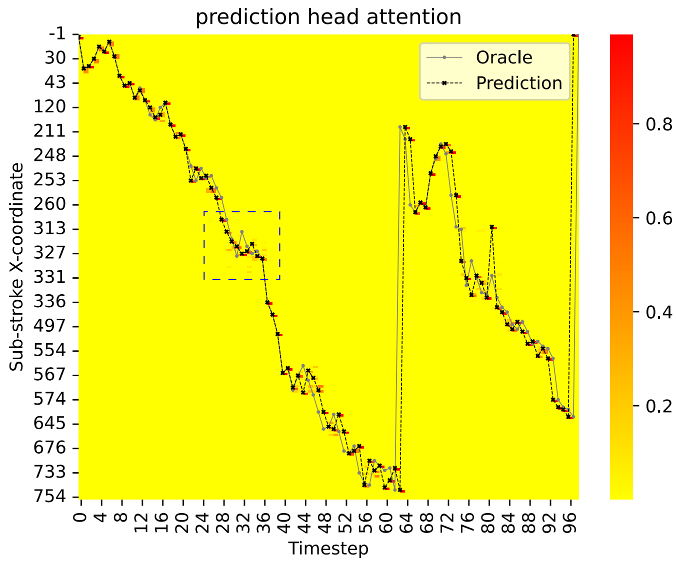
Fig. 7: Inference results of [4] and our approach on CROHME datasets.

we need this transfer, in the form of a fine-tuning step, for new application domains. The existing online databases of handwriting in a multitude of languages and free-form charts can be exploited to train the system in order to generalize to all content types.

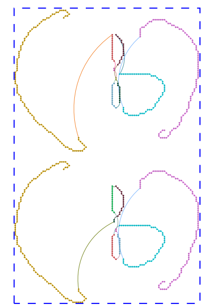
We focus our study application on offline images of at most 100 sub-strokes. Full offline documents can attain upwards of 6000 sub-strokes. Further research efforts are necessary to up-scale our sub-stroke ordering transformer to the document level. In fact, it presents two difficulties, firstly longer and more complex temporal dependencies to model. Secondly, the memory bottleneck of the quadratic multi-head attention needs to be addressed.

7 Conclusion

In this paper, we presented a novel sub-stroke level transformer approach to recover online from offline handwriting. Our approach consists of two steps: First, we embed the sub-strokes set, extracted from the inferred skeleton, using a sub-stroke encoding transformer (SET). The sub-strokes embeddings are ordered using a sub-strokes ordering Transformer (SORT) which also predicts the pen



(a) Output attention heatmap during inference.



(b) Top network's prediction and bottom oracle's order.

Fig. 8: (a) Attention heatmap of the SORT decoder output layer for the equation 7d. The y-axis is the memory sub-strokes sorted from left to right (with the first point) for illustration purposes only. Network predictions (see section 3.6) as well as the oracle answers are plotted on top of the heatmap. The *eos* sub-stroke is here indicated by a -1. (b) The divergence between the inferred sub-stroke order and the oracle's order, for timesteps from 24 to 40.

state. In contrast to other data-driven approaches, SORT is trained in a guided attention manner and is able to accurately string together the original sub-strokes rather than regressing a simplified approximation of the online. Our method's performance stands out when compared to the state-of-the-art on Latin words and Math equations. In future work, we would like to extend our system to full documents thus enabling a powerful combination of offline note-taking and seamless online editing.

8 Acknowledgements

We would like to express our gratitude to Robin Mélinand for his invaluable feedback and suggestions for this article.

References

1. Aksan, E., Deselaers, T., Tagliasacchi, A., Hilliges, O.: CoSE: Compositional stroke embeddings. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. pp. 10041–10052. NIPS'20, Curran Associates Inc., Red Hook, NY, USA (Dec 2020), https://proceedings.neurips.cc/paper/2020/file/723e8f97fde15f7a8d5ff8d558ea3f16-Paper.pdf
2. Archibald, T., Poggemann, M., Chan, A., Martinez, T.: TRACE: A Differentiable Approach to Line-Level Stroke Recovery for Offline Handwritten Text. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) Document Analysis and Recognition – ICDAR 2021, vol. 12823, pp. 414–429. Springer International Publishing, Cham (2021). https://doi.org/10.1007/978-3-030-86334-0_27
3. Bhunia, A.K., Bhowmick, A., Bhunia, A.K., Konwer, A., Banerjee, P., Roy, P.P., 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 (Aug 2018). https://doi.org/10.1109/ICPR.2018.8546093
4. Chan, C.: Stroke Extraction for Offline Handwritten Mathematical Expression Recognition. IEEE Access **8**, 61565–61575 (2020). https://doi.org/10.1109/ACCESS.2020.2984627
5. Doermann, D., Intrator, N., Rivin, E., Steinherz, T.: Hidden loop recovery for handwriting recognition. In: Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition. pp. 375–380 (Aug 2002). https://doi.org/10.1109/IWFHR.2002.1030939
6. Egiazarian, V., Voynov, O., Artemov, A., Volkhonskiy, D., Safin, A., Taktasheva, M., Zorin, D., Burnaev, E.: Deep Vectorization of Technical Drawings. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M. (eds.) Computer Vision – ECCV 2020, vol. 12358, pp. 582–598. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-58601-0_35
7. Guo, Y., Zhang, Z., Han, C., Hu, W., Li, C., Wong, T.T.: Deep Line Drawing Vectorization via Line Subdivision and Topology Reconstruction. Computer Graphics Forum **38**(7), 81–90 (2019). https://doi.org/10.1111/cgf.13818
8. Ha, D., Eck, D.: A neural representation of sketch drawings. In: ICLR 2018 (2018), https://openreview.net/pdf?id=Hy6GHpkCW, 2018
9. Holten, D., van Wijk, J.J.: A user study on visualizing directed edges in graphs. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 2299–2308. CHI '09, Association for Computing Machinery, New York, NY, USA (Apr 2009). https://doi.org/10.1145/1518701.1519054
10. Jager, S.: Recovering writing traces in off-line handwriting recognition: Using a global optimization technique. In: Proceedings of 13th International Conference on Pattern Recognition. vol. 3, pp. 150–154 vol.3 (Aug 1996). https://doi.org/10.1109/ICPR.1996.546812
11. Ji, B., Chen, T.: Generative Adversarial Network for Handwritten Text (Feb 2020)
12. Mahdavi, M., Zanibbi, R., Mouchere, H., Viard-Gaudin, C., Garain, U.: ICDAR 2019 CROHME + TFD: Competition on Recognition of Handwritten Mathematical Expressions and Typeset Formula Detection. In: 2019 International Conference on Document Analysis and Recognition (ICDAR). pp. 1533–1538 (Sep 2019). https://doi.org/10.1109/ICDAR.2019.00247
13. Marti, U.V., Bunke, H.: The IAM-database: An English sentence database for offline handwriting recognition. International Journal on Document Analysis and Recognition **5**(1), 39–46 (Nov 2002). https://doi.org/10.1007/s100320200071

14. Mohamed Moussa, E., Lelore, T., Mouchère, H.: Applying End-to-End Trainable Approach on Stroke Extraction in Handwritten Math Expressions Images. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) Document Analysis and Recognition – ICDAR 2021. pp. 445–458. Lecture Notes in Computer Science, Springer International Publishing, Cham (2021). https://doi.org/10.1007/978-3-030-86334-0_29
15. Mohamed Moussa, E., Lelore, T., Mouchère, H.: Point to Segment Distance DTW for Online Handwriting Signals Matching:. In: Proceedings of the 12th International Conference on Pattern Recognition Applications and Methods. pp. 850–855. SCITEPRESS - Science and Technology Publications, Lisbon, Portugal (2023). https://doi.org/10.5220/0011672600003411
16. Nguyen, V., Blumenstein, M.: Techniques for static handwriting trajectory recovery: A survey. In: Proceedings of the 9th IAPR International Workshop on Document Analysis Systems. pp. 463–470. DAS '10, Association for Computing Machinery, New York, NY, USA (Jun 2010). https://doi.org/10.1145/1815330.1815390
17. Qiao, Y., Nishiara, M., 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 (Nov 2006). https://doi.org/10.1109/TPAMI.2006.216
18. Seki, Y.: Online and Offline Data Collection of Japanese Handwriting. In: 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW). vol. 8, pp. 13–18 (Sep 2019). https://doi.org/10.1109/ICDARW.2019.70135
19. Simo-Serra, E., Iizuka, S., Ishikawa, H.: Mastering Sketching: Adversarial Augmentation for Structured Prediction. ACM Transactions on Graphics **37**(1), 11:1–11:13 (Jan 2018). https://doi.org/10.1145/3132703
20. Viard-Gaudin, C., Lallican, P.M., Knerr, S., 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 (Sep 1999). https://doi.org/10.1109/ICDAR.1999.791823
21. Vinciarelli, A., Perone, M.: Combining online and offline handwriting recognition. In: Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings. pp. 844–848 (Aug 2003). https://doi.org/10.1109/ICDAR.2003.1227781
22. Zhang, J., Du, J., Dai, L.: Track, Attend, and Parse (TAP): An End-to-End Framework for Online Handwritten Mathematical Expression Recognition. IEEE Transactions on Multimedia **21**(1), 221–233 (Jan 2019). https://doi.org/10.1109/TMM.2018.2844689
23. Zhang, T.Y., Suen, C.Y.: A fast parallel algorithm for thinning digital patterns. Communications of the ACM **27**(3), 236–239 (Mar 1984). https://doi.org/10.1145/357994.358023
24. Zhao, B., Yang, M., Tao, J.: Pen Tip Motion Prediction for Handwriting Drawing Order Recovery using Deep Neural Network. In: 2018 24th International Conference on Pattern Recognition (ICPR). pp. 704–709 (Aug 2018). https://doi.org/10.1109/ICPR.2018.8546086
25. Zhao, W., Gao, L.: CoMER: Modeling Coverage for Transformer-Based Handwritten Mathematical Expression Recognition. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision – ECCV 2022. pp. 392–408. Lecture Notes in Computer Science, Springer Nature Switzerland, Cham (2022). https://doi.org/10.1007/978-3-031-19815-1_23