applied sciences

MDPI

*Article*

# Transcription Alignment of Historical Vietnamese Manuscripts without Human-Annotated Learning Samples

**Anna Scius-Bertrand [1,2,*], Michael Jungo [1], Beat Wolf [1], Andreas Fischer [1,3] and Marc Bui [2]**

[1] iCoSys, University of Applied Sciences and Arts Western Switzerland, 1700 Fribourg, Switzerland; michael.jungo@hefr.ch (M.J.); beat.wolf@hefr.ch (B.W.); andreas.fischer@hefr.ch (A.F.)
[2] Ecole Pratique des Hautes Etudes, PSL, 75014 Paris, France; marc.bui@ephe.sorbonne.fr
[3] DIVA, University of Fribourg, 1700 Fribourg, Switzerland
* Correspondence: anna.scius-bertrand@hefr.ch

**Abstract:** The current state of the art for automatic transcription of historical manuscripts is typically limited by the requirement of human-annotated learning samples, which are are necessary to train specific machine learning models for specific languages and scripts. Transcription alignment is a simpler task that aims to find a correspondence between text in the scanned image and its existing Unicode counterpart, a correspondence which can then be used as training data. The alignment task can be approached with heuristic methods dedicated to certain types of manuscripts, or with weakly trained systems reducing the required amount of annotations. In this article, we propose a novel learning-based alignment method based on fully convolutional object detection that does not require any human annotation at all. Instead, the object detection system is initially trained on synthetic printed pages using a font and then adapted to the real manuscripts by means of self-training. On a dataset of historical Vietnamese handwriting, we demonstrate the feasibility of annotation-free alignment as well as the positive impact of self-training on the character detection accuracy, reaching a detection accuracy of 96.4% with a YOLOv5m model without using any human annotation.

**Keywords:** transcription alignment; object detection; self-training; YOLO; chu nom characters



**Citation:** Scius-Bertrand, A.; Jungo, M.; Wolf, B.; Fischer, A.; Bui, M. Transcription Alignment of Historical Vietnamese Manuscripts without Human-Annotated Learning Samples. *Appl. Sci.* **2021**, *11*, 4894. https://doi.org/10.3390/app11114894

Academic Editors: Byron Leite Dantas Bezerra, Donato Impedovo and Carlos-D. Martínez Hinarejos

Received: 5 April 2021
Accepted: 10 May 2021
Published: 26 May 2021



## 1. Introduction

To preserve and access our cultural heritage, the digitization of historical manuscripts is an important goal for libraries all around the world. Scanning the documents and indexing them with meta-information about author, date, place, etc. is a first step towards this goal. Afterwards, automatic document analysis and recognition is needed to extract texts, illustrations, signatures, stamps, etc. from the scanned page images and make them amenable to searching, browsing, indexing, and linking similar to websites on the Internet. In recent decades, a steady advance in research has made it possible to analyze even highly degraded manuscripts written in ancient languages and scripts [1].

The majority of current methods are based on machine learning and thus have a fundamental limitation —the need for human-annotated learning samples to train the document analysis systems. In the context of automated reading, such learning samples may consist for example of bounding boxes drawn around text lines together with their machine-readable text. Considering the wide variety of historical documents, it is often necessary to manually annotate dozens or hundreds of pages, only to transcribe a few thousand similar pages afterwards with an automatic system.

Automatic transcription alignment [2] has been suggested as a promising approach to cope with this limitation for cases where scholars have already created a transcription. In such a case, aligning the words in the image with the words in the transcription not only facilitates browsing of the scanned manuscript but also provides the ground truth information needed to train an automated reading system. Several methods have been put forward to tackle this challenge (see Section 2), some of them based on heuristics

avoiding machine learning altogether and others using weakly trained alignment systems, i.e., systems that require only a few labeled samples for the alignment task, which is simpler than the reading task. In either case, the alignment methods have to provide a solution for two distinct problems. First, for segmenting the image into text elements, and secondly, for aligning the image segments with the text.

In this article, we propose an alignment method, which solves the segmentation problem by means of fully convolutional object detection and the alignment problem by means of clustering of the detected bounding boxes. For training the object detection system, we use purely synthetic learning samples based on a gray text background and a printed font. Subsequently, the synthetic detection system is applied to a real manuscript and aligns the transcription. Finally, self-training is performed on the alignment results without human supervision in order to adapt from the gray background to the real page background and from the printed font to the handwriting style. The resulting adapted detection system is an ideal starting point for further document analysis steps, such as keyword spotting and transcription.

The proposed transcription alignment method has the following key properties:

- *Segmentation-free*. The method can be applied directly to the scanned page without image preprocessing, paragraph, line, or word segmentation.
- *Annotation-free*. No human-annotated learning sample is required, which is the main contribution of the proposed method.
- *Learning-based*. The segmentation problem is addressed with machine learning, which tends to be more robust to variations in the page background and the handwriting style when compared with heuristic methods.

To the best of our knowledge, the proposed approach is one of the first that combines the three properties, which greatly facilitate transcription alignment. Please note, however, that a font of the script under consideration is required. Furthermore, the clustering algorithm used to solve the alignment problem has been specifically designed for the dataset at hand, and would need to be adapted when dealing with other types of manuscripts.

The method is experimentally evaluated on a dataset of historical Vietnamese handwriting [3]. The use of this dataset is inspired by recent work in the literature, which has shown that object detection is effective for detecting characters in historical Chinese documents [4] and that the use of synthetic printed characters is effective for training fully convolutional image segmentation for the Nom characters used in historical Vietnamese manuscripts [3]. In our experiments, we test both the feasibility of transcription alignment without human annotation and the impact of self-training with the alignment results on the character detection accuracy.

The remainder of the article is structured as follows. Related work is discussed in Section 2, the dataset of Vietnamese handwriting is described in Section 3, the object detection method used is elaborated in Section 4, the proposed alignment algorithms is introduced in Section 5, and experimental evaluations are presented in Section 6. Finally, we draw some conclusions and provide an outlook to future work in Section 7.

## 2. Related Work

### 2.1. Historical Handwriting Recognition

The interest in historical handwriting analysis and recognition has increased substantially in the last decade [1]. Numerous research datasets have been created for different scripts and languages, including for example the English George Washington database [5], the Bangla CMATERdb1 database [6], the Spanish ESPOSALLES database [7], the Arabic VML-HD database [8], the Chinese CASIA-AHCDB [9], and the Swedish ARDIS database [10]. Additionally, several competitions have been held to compare state-of-the-art approaches to text line and word segmentation [11], handwritten keyword spotting [12], and handwriting recognition [13], to name just a few. The current state of the art is in large parts based on deep convolutional neural networks, with attention mechanisms playing an important role in recent architectures [14,15]. Most methods for reading handwriting

are segmentation-based, i.e., focusing on pre-segmented paragraphs, text lines, or words, and are trained with human-annotated handwriting samples, typically text line images together with their corresponding transcription.

The creation of training samples for historical handwriting is a time-consuming and costly process. For modern handwriting, standard benchmark datasets such as the IAM database [16] have been obtained by asking writers to copy a given text by hand into special forms that facilitate the automatic extraction of handwriting images together with their corresponding transcription. For historical handwriting, however, time-consuming manual annotations of scanned manuscripts are needed. They typically consist of bounding boxes or polygons around text elements, together with their machine-readable transcription, which can often only be provided by experts in the case of ancient languages.

### 2.2. Transcription Alignment

A large number of historical manuscripts have already been transcribed by scholars, but in most cases the transcriptions are not aligned with the scanned manuscript images, i.e., it is not known where on the page the different words and characters are located. Transcription alignment aims to establish such an alignment automatically, which greatly facilitates the creation of historical handwriting datasets. Several alignment methods have been suggested for Latin text written in text lines. Avoiding machine learning entirely, *heuristic transcription alignment approaches* include the method proposed by Leydier et al. [17] for medieval Latin manuscripts, which first performs a gradient-based line segmentation, followed by feature matching between the line images and the Unicode transcription using dynamic programming. Another example is the method proposed by Rabaev et al. [18] for historical Hebrew manuscripts, which employs scale-space anisotropic smoothing for segmenting the lines and dynamic programming to match the connected components of the line images with a sequence of synthetic prototype characters.

Heuristic methods are limited in their generalizability when dealing with a certain variability of page layouts, page backgrounds, and handwriting styles. Therefore, several *learning-based transcription alignment approaches* have been investigated as well, including the hidden Markov model (HMM)-based approach proposed by Fischer et al. [19] for medieval Latin manuscripts, which is based on a heuristic line segmentation with seam carving, followed by an HMM-based forced alignment with the Viterbi algorithm. This method specifically addressed the problem of discrepancies between the text visible in the image and the human transcription, especially with a view to abbreviations that are frequent in medieval Latin texts. In a similar method proposed by Romero et al. [20], the heuristic line segmentation is further replaced with an HMM-based approach. Moving from HMM to convolutional approaches, Chammas et al. [21] use convolutional neural network (CNN) features in combination with long short-term memory (LSTM) cells and connectionist temporal classification (CTC) to align text line images with their transcription. In their work, line segmentation is achieved heuristically by means of contour distribution analysis. Recently, Ziran et al. [22] have proposed the use of fully convolutional object detection for aligning early printed Latin documents. To train a Faster R-CNN architecture, lines and words are first segmented heuristically using binarization and projection profiles. The object detection results are then aligned with the transcription using dynamic programming.

For modern Chinese handwriting, a transcription alignment method is proposed by Yin et al. [23], using a minimal spanning tree for line segmentation and a statistical classifier, supported by geometric context, for alignment using dynamic programming. Please note that all works mentioned are based on some form of dynamic programming to align part-of-image sequences with text sequences.

When compared with the alignment approaches mentioned, our method is most closely related to that of Ziran et al. [22], because we also base our alignment on object detection. However, we do not require an initial image preprocessing step that segments the scanned page image into text blocks, lines, or words. Instead, we train the object detection method with synthetic page images and apply it to entire scanned pages in order

to localize text elements before alignment. Another difference is the alignment algorithm itself, which is not sequential in our case but instead based on two-dimensional clustering into columns and rows.

### 2.3. Training with Printed Nom Characters

With respect to the use of printed training data for detecting Nom characters in historical Vietnamese handwriting, the proposed approach is closely related to the work of Nguyen et al. [3], where the authors have demonstrated how pre-training a U-Net on printed data leads to good character detection results. For training, they generate thousands of synthetic pages with randomly distorted characters from several Nom fonts printed in different resolutions on a white background. Convex hulls of the characters are used as ground-truth for training a U-Net. To adapt their system to specific historical manuscripts, *a subset of the manuscripts is annotated by humans and used to fine-tune the U-Net*. At testing time, the results of the U-Net-based semantic segmentation are post-processed with a watershed algorithm to obtain individual characters.

When compared with [3], we use a similar procedure to create a synthetic training set based on printed Nom characters (see Section 5.1). Bounding boxes around the characters are used as ground-truth for training a YOLO-based object detection network (see Section 4). However, *we do not use any human-annotated learning samples to fine-tune the YOLO network*. Instead, we perform transcription alignment with the synthetically trained network and a clustering-based algorithm (see Section 5.2) in order to automatically generate training data from an independent set of manuscripts for fine-tuning. Another difference is that, at testing time, no post-processing is needed for obtaining individual characters. Instead, the YOLO network provides directly character bounding boxes as output.

## 3. Dataset of Historical Vietnamese Handwriting

The dataset considered for experimental evaluation in this article is a collection of manuscripts written in Chu Nom used in Vietnam over 1000 years from the 10th to the 20th century, before it was replaced with a Latin-based alphabet. The script has Chinese origins but significantly extends the traditional Chinese writing system. Fewer and fewer people are still able to read Nom characters and therefore cultural heritage preservation is of a certain urgency. The Vietnamese Nom Preservation Foundation (VNPF) (http://www.nomfoundation.org/ accessed on 25 May 2021) has collected a comprehensive collection of scanned manuscripts together with their transcriptions and also created the *Nom Na Tong Light* font, which currently includes nearly 30,000 Nom characters in Unicode. For transcription alignment, we consider five different version of the *Tale of Kieu* (1866, 1870, 1871, 1872, 1902) and the story of *Luc Van Tien*, which are published online by the Vietnamese Nom Preservation Foundation and were kindly made available to us for research purposes. Altogether, they include 899 scanned images and their transcriptions in Unicode, which contain column breaks but no information where on the scanned page the characters and columns are located. Please note that the text is read from top to bottom, right to left.

For evaluating the character detection system, we consider an independent test set of 47 manuscript images used by Nguyen et al. [3] with a manual ground truth for each character bounding box, which were kindly shared with us by the authors.

Figure 1a illustrates both parts of the dataset. Please note that they differ significantly in terms of page layout, page background, page border, and handwriting style. Our proposed method will be applied, first, to align the 899 pages using only synthetic printed characters from the Nom font for training and, secondly, to self-train the detection system on the aligned pages in order to improve the character detection accuracy. To assess the improvement, we measure the detection accuracy on the 47 test pages before and after self-training.

(**a**) Example images of the manuscripts considered for transcription alignment. They include the *Tale of Kieu* and *Luc Van Tien* made available online by the Vietnamese Nom Preservation Foundation. The transcriptions include the main columns, separated by a column break, but not the additional characters in the middle or the half-visible characters on the page border

(**b**) Example images of the manuscripts considered for character detection. They include the 47 test images used in [3]

**Figure 1.** Dataset of historical Vietnamese handwriting.

During the entire process, no human annotation is required for training the system, i.e., no bounding boxes have to be drawn around characters on the scanned pages. However, of course, we integrate the immense work of the scholars that was necessary to create the transcriptions and the Nom font.

## 4. Fully Convolutional Character Detection

In computer vision, object detection refers to the task of locating and identifying objects within images. This is most commonly associated with everyday items in a natural context, but the same idea and technology can be applied to detect text in images. In this context, it is much more common that there are many smaller objects rather than a few larger ones. Consequently, each character occupies a much smaller part of the overall image, requiring larger input resolutions, otherwise the characters become indistinguishable or might vanish entirely. One-stage object detectors are particularly well suited for this, as they not only tend to be more efficient, but perform dense predictions, i.e., for every pixel of the feature maps a prediction is conducted. This means that each pixel is part of the prediction, regardless of how many bounding boxes are actually present, and therefore, having more bounding boxes does not increase the complexity.

In this article, we use the YOLO [24] (You Only Look Once) architecture to detect characters, which has pioneered one-stage object detection and showed that competitive results can be achieved with a single convolutional neural network. Its simplicity and efficiency sparked the interest around this family of models. Other one-stage object detectors, such as SSD [25] and RetinaNet [26], followed a very similar approach to YOLO and improved upon it. These models inspired further progress in this domain, including the evolution of YOLO, which remained competitive over the years in terms of both speed and accuracy, making it one of the most popular and widely used object detection models. In the following, we describe its evolution over time and its current state of development.

In the original version of YOLO [24] the images have been divided in a $7 \times 7$ grid of cells, where each cell would be assigned a single class and only predict up to two boxes. This was a major limitation in regards to small objects, as there could potentially be more than two objects per cell or two objects of a different class, which could not be detected appropriately.

YOLOv2 [27] addressed these shortcomings with three major changes. First, the input resolution has been increased in order to avoid losing too much information about small objects. Secondly, multi-scale training was employed to make the model more robust across different scales of objects, because not all scales are represented equally. Lastly, the biggest change was the adaptation of anchor boxes, introduced by the two-stage object detector Faster R-CNN [28].

Anchor boxes are predefined bounding boxes, which are used as a prior and further refined to obtain to the precise bounding box. These anchor boxes need to be chosen carefully, because they determine whether a particular starting point should be considered a candidate for an object, usually based on the intersection over union (IoU) during training, and then proceed to the refinement of the bounding box. To obtain the best coverage, multiple anchor sizes and aspect ratios are selected, which should be representative of the bounding boxes that can be found in the training data.

For the next iteration, YOLOv3 [29], no drastic changes have been made, but a few design changes resulted in an incremental improvement of the model. The most notable addition was the integration of a Feature Pyramid Network (FPN) [30], commonly referred to as the *neck* in the network architecture. In the FPN, multiple feature maps are generated at different scales and successive feature maps flow back into earlier stages. As a result, the feature maps across all levels become semantically stronger, making the model more robust to different scales. YOLOv3 uses three levels in the FPN and therefore also predicts bounding boxes at these three scales.

With the general architecture being established, depicted in Figure 2, YOLOv4 [31] focused on comparing numerous existing building blocks in order to find the best possible configuration, which consists of the following: A variation of the Cross Stage Partial Network (CSPNet) [32] as the backbone, a modified Path Aggregation Network (PAN) [33] as the neck, and the existing anchor-based head of YOLOv3.

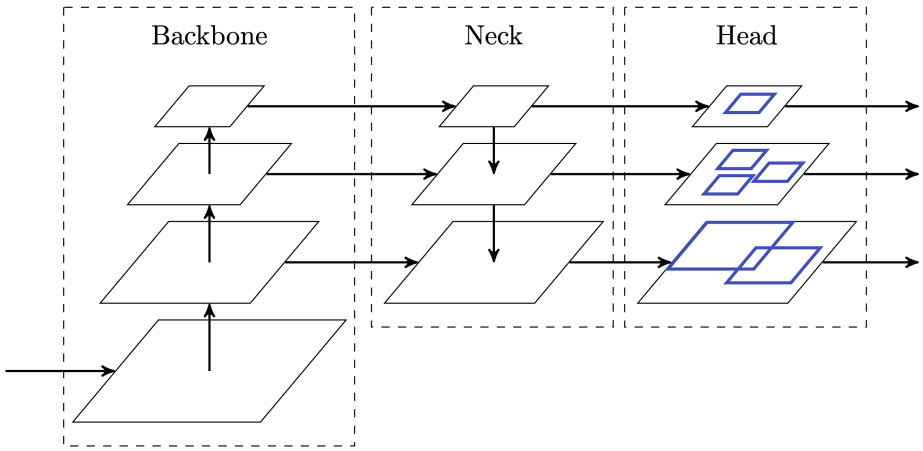



**Figure 2.** The model architecture of YOLO, where the backbone extracts features from an image, the neck enhances feature maps by combining different scales, and the head predicts the bounding boxes.

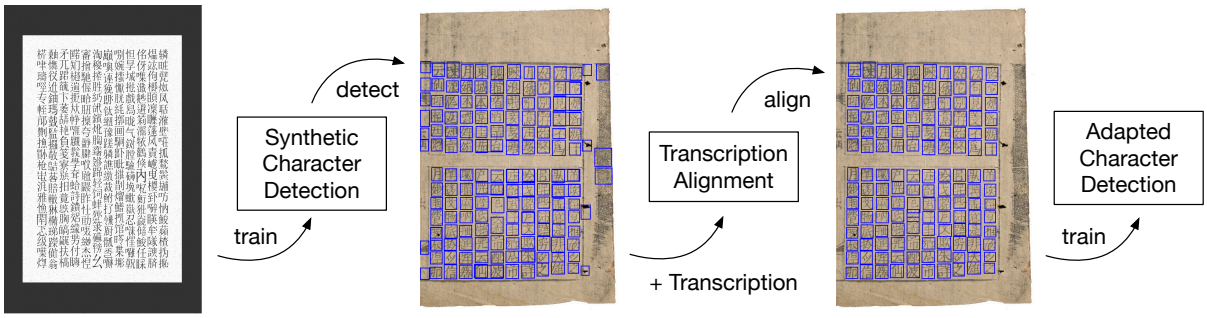
We have decided to opt for YOLOv5 [34], an accessible version with various improvements, including in terms of convenience, such as automatic learning of anchor boxes from the distribution of the bounding boxes in the specified dataset with a k-means clustering and genetic algorithm. Contrary to its name, it is not the successor of YOLOv4 but rather a port of YOLOv3 to PyTorch [35], which has been improved independently from YOLOv4 and many changes found in YOLOv4 have later been integrated into it as well.

## 5. Annotation-Free Transcription Alignment

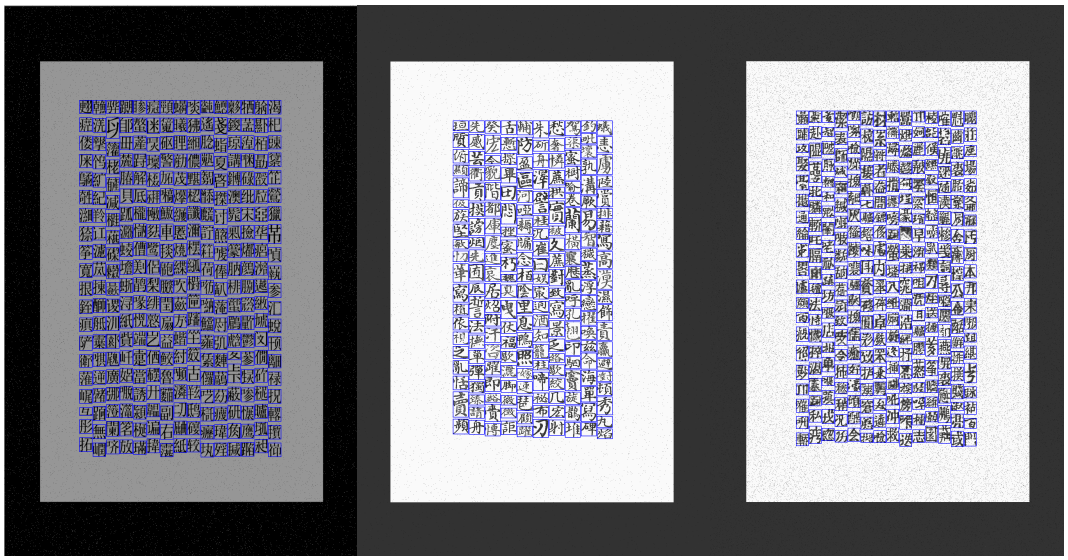
The proposed method for handwritten transcription alignment without human annotation is based on two components, a character detection system (see Section 4) and a transcription alignment algorithm (introduced below).

Figure 3a provides an overview of the method. First, the Nom font is used to generate simple synthetic pages without a particular parchment background. They are used to train an initial *synthetic* character detection system, which is then applied to the handwriting dataset (see Section 3). Afterwards, the detection result is aligned with the existing transcription using unsupervised clustering techniques. Finally, without human correction, the resulting annotations are used to train an *adapted* character detection system, i.e., continuing to train the synthetic system with the self-annotated handwriting data. This last self-training step aims to perform a transfer from printed character detection to handwritten character detection using the alignment results.

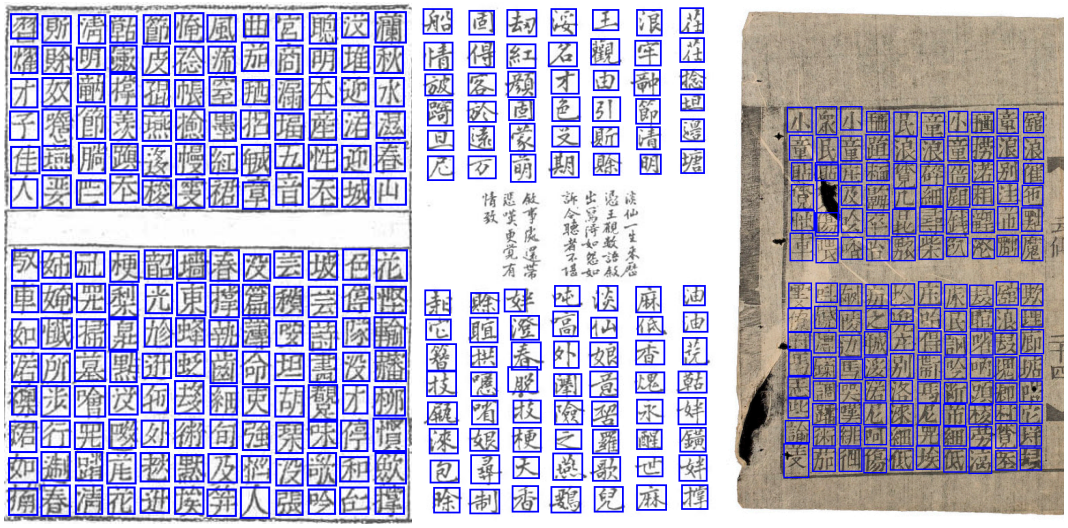
In the following, we provide more details on the synthetic document generation, introduce the alignment algorithm, and discuss the performance measures used for assessing the alignment quality.




(**a**) Method overview. First, synthetically generated pages are used to train a character detection system, which is then applied to the manuscripts. Secondly, the detected bounding boxes are aligned with an existing transcription. Finally, the alignment results are used to adapt the character detection system to the manuscripts



(**b**) Synthetic document generation using printed characters of the font made available online by the Vietnamese Nom Preservation Foundation (**left**), characters of the CASIA-AHCDB [9] (**middle**), and Nom characters obtained from transcription alignment (**right**)



(**c**) Transcription alignment on the *Tale of Kieu* and *Luc Van Tien*. The characters not belonging to the main columns are correctly excluded from the result (**middle**). On closer inspection of the page from *Luc Van Tien* (**right**), some mistakes in the bounding box positions are visible

**Figure 3.** Annotation-free transcription alignment.

### *5.1. Synthetic Document Generation*

Synthetic documents are generated with a gray main text area, surrounded by a black border. Characters are randomly chosen from a dictionary, e.g., all 26,969 characters of the Nom font, and are added with equal character width to the main text area with variable outside border and number of columns, in order to generate training data at different scales. Afterwards, several distortions are applied to the document, including shift, Gaussian blur, salt and pepper noise, and changes in brightness.

In addition to printed characters, we also consider two types of handwritten characters. First, traditional Chinese characters from the CASIA-AHCDB [9] and, secondly, Nom characters obtained from our automatic transcription alignment. Examples for all three types of characters are shown in Figure 3b.

An important aspect of data generation for object detection is the choice of bounding boxes. We use tight bounding boxes that touch the characters by applying a border shrinking procedure to the printed or handwritten character images, which ignores small foreground elements. The same procedure will also be considered in the context of performance evaluation (see Section 5.3).

Algorithm 1 details the shrinking process. First, the image is transformed to grayscale and binarized using a global Otsu threshold, which minimizes the intra-class variance for both the image background and the character foreground (Line 1). Afterwards the borders are moved inside, as long as the cumulative number of foreground pixels does not exceed a threshold $\tau$ (Line 4). The borders are set to the last zero-pixel column or row encountered before the threshold is reached (Line 6).

**Algorithm 1** Character bounding box

**Require:**
$R[n, m]$: RGB character image with $n$ columns and $m$ rows
$\tau$: minimum number of foreground pixels
**Ensure:**
$b_{left}$, $b_{right}$, $b_{top}$, $b_{bottom}$: bounding box coordinates
1: $B \leftarrow$ Otsu(Gray($R$))
2: $P_h$, $P_v \leftarrow$ horizontal and vertical projection profiles of binary image $B$
3: $b_{left} = 0$; $column = 0$; $sum = 0$
4: **while** $column < n$ and $sum < \tau$ **do**
5: **if** $P_v[column] == 0$ **then**
6: $b_{left} \leftarrow column$; $sum \leftarrow 0$
7: **else**
8: $sum \leftarrow sum + P_v[column]$
9: **end if**
10: $column \leftarrow column + 1$
11: **end while**
12: similar for $b_{right}$, $b_{top}$, and $b_{bottom}$
13: **return** $b_{left}$, $b_{right}$, $b_{top}$, $b_{bottom}$

Because the proposed method does not take into account human annotations for optimization of system parameters, reasonable defaults have to be chosen. In this article, we use $\tau = 10$ as default value. Figure 3b shows the resulting tight bounding boxes.

### *5.2. Transcription Alignment*

After training a fully convolutional object detection system on synthetic training data, it is applied to the dataset of historical Vietnamese handwriting. For each scanned page, the bounding boxes $\mathcal{B}$ of the detected characters and the existing transcription $T[n, m]$ with $n$ columns and $m$ rows are provided as input to the proposed alignment method, which is described in Algorithm 2. $T[i, j]$ with $1 \leq i \leq n$, $1 \leq j \leq m$, denotes the $j$th character of the $i$th column in the transcription, according to the column breaks present in the transcription.

**Algorithm 2** Transcription alignment

**Require:**
$\mathcal{B}$: set of detected character bounding boxes
$T[n, m]$: transcription with $n$ columns and $m$ rows
$\sigma_{size}$: maximum size deviation
$\sigma_{overlap}$: maximum overlap
$\sigma_{border}$: minimum border distance
**Ensure:**
$\mathcal{A}$: set of aligned character bounding boxes
1: $b_m \leftarrow median(\mathcal{B})$
2: $\mathcal{R} \leftarrow remove_{size}(\mathcal{B}, b_m, \sigma_{size})$
3: $\mathcal{R} \leftarrow remove_{overlap}(\mathcal{R}, b_m, \sigma_{overlap})$
4: $\mathcal{R} \leftarrow remove_{border}(\mathcal{R}, \sigma_{border})$
5: $columns \leftarrow KMeans_x(\mathcal{R}, k = n)$
6: $column_m \leftarrow median(columns)$
7: $rows \leftarrow KMeans_y(\mathcal{R}, k = m)$
8: $row_m \leftarrow median(rows)$
9: $\mathcal{A} = \{\}$
10: **for** each column index i **do**
11: **for** each row index j **do**
12: $(x, y) \leftarrow grid\_location(column_m, row_m, i, j)$
13: **if** $has\_nearest(\mathcal{R}, x, y)$ **then**
14: $box \leftarrow nearest(\mathcal{R}, x, y)$
15: **else**
16: $box \leftarrow grid\_box(column_m, row_m, i, j)$
17: **end if**
18: $box.character \leftarrow T[i, j]$
19: $\mathcal{A} \leftarrow \mathcal{A} \cup box$
20: **end for**
21: **end for**
22: **return** $\mathcal{A}$

First, the median bounding box $b_m$ is determined with respect to the larger box side $max(width,\ height)$ (Line 1). Afterwards, the following three types of outliers are removed from $\mathcal{B}$ (Lines 2–4):

- Boxes that deviate more than $\sigma_{size}$ in both width and height from $b_m$.
- Boxes that overlap more than $\sigma_{overlap}$ with another box and have the smaller intersection over union (IoU) with $b_m$ when compared with the other box.
- Boxes that are less than $\sigma_{border}$ pixels away from the image border.

Next, k-means clustering is applied to the remaining $\mathcal{R}$ boxes using the $x$ center coordinates and $k = n$, in order to obtain columns (Line 5). Among all columns that have exactly $m$ elements and are thus expected to be correctly identified, the median $column_m$ is determined with respect to the sum of squared distances of the $y$ center coordinates (Line 6). Similarly the median $row_m$ is determined using k-means clustering of $\mathcal{R}$ in vertical direction (Lines 7 and 8).

The median column and the median row have a common crossing box $b_c$ and span a grid of $n \times m$ cells. To build the set of aligned boxes $\mathcal{A}$, each cell is visited once (Lines 9–11). In Line 12, the $(x, y)$ center position is calculated as:

$$x = column_m[j].x - (b_c.x - row_m[i].x) \quad (1)$$

$$y = column_m[j].y - (b_c.y - row_m[i].y) \quad (2)$$

If there is one or several boxes in $\mathcal{R}$ that include this center point (Line 13), the closest such box is selected as the alignment result for the cell (Line 14). Otherwise, the translated box $column_m[j]$ of the median column is used (Line 16), ensuring that each cell has an alignment result. This procedure is useful to fill gaps in the detection result around the position $(x, y)$.

Finally, the character of the bounding box is set to the corresponding character from the transcription and the box is added to $\mathcal{A}$ (Lines 18–19), which is returned as the final result. Figure 3c illustrates the resulting alignment for $\sigma_{size} = 0.2$, $\sigma_{overlap} = 0.1$, and $\sigma_{border} = 5$, which are used as default values in this article.

### 5.3. Performance Measures

For assessing the quality of the character detection, we consider the standard intersection over union (IoU) measure

$$IoU(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{3}$$

with respect to ground truth $A$ and detection result $B$. Since the IoU is highly sensitive to small changes in the width and height of the bounding boxes, we apply the box shrinking procedure detailed in Algorithm 1 to the detection results, in order to obtain tight boxes for performance evaluation. This holds true for all reported results in this article if not stated otherwise.

Because the number of ground truth boxes may be different from the number of detected boxes, we first establish an optimal assignment between the ground truth and the detection results. For this purpose, we solve a linear sum assignment problem with the IoU as the underlying matching cost. This results in the following three types of boxes:

- Matching pairs $\mathcal{M} = \{(A, B)\}$
- Deletions $\mathcal{D} = \{(A, \epsilon)\}$, i.e., unassigned ground truth boxes
- Insertions $\mathcal{I} = \{(\epsilon, B)\}$, i.e., unassigned detection results

Furthermore, we distinguish successful assignments $\mathcal{M}^+$ and failed assignments $\mathcal{M}^-$

$$\mathcal{M}^+ = \{(A, B) : IoU(A, B) \geq 0.5\} \tag{4}$$
$$\mathcal{M}^- = \{(A, B) : IoU(A, B) < 0.5\} \tag{5}$$

with $\mathcal{M}^+ \cup \mathcal{M}^- = \mathcal{M}$. Based on these distinctions, and with the total number of assignments $N = |\mathcal{M}| + |\mathcal{D}| + |\mathcal{I}|$, we calculate the mean IoU (IoU), precision (P), recall (R), $F_1$ score (F1), and character detection accuracy (Acc) as follows:

$$IoU = \frac{\sum_{(A,B)\in\mathcal{M}} IoU(A, B)}{N} \tag{6}$$
$$P = \frac{|\mathcal{M}^+|}{|\mathcal{M}| + |\mathcal{I}|} \tag{7}$$
$$R = \frac{|\mathcal{M}^+|}{|\mathcal{M}| + |\mathcal{D}|} \tag{8}$$
$$F1 = \frac{2PR}{P + R} \tag{9}$$
$$Acc = \frac{N - |\mathcal{M}^-| - |\mathcal{D}| - |\mathcal{I}|}{N} \tag{10}$$

Please note that our definition of character detection accuracy is closely related to the character recognition accuracy commonly used for handwriting recognition. It takes into account substitution errors ($\mathcal{M}^-$), deletion errors ($|\mathcal{D}|$), and insertion errors ($|\mathcal{I}|$). Because the number of insertions errors is not limited, the accuracy can also be negative.

## 6. Experimental Evaluation

Two goals are pursued for experimental evaluation. First, to assess the feasibility of transcription alignment in the absence of human-annotated learning samples and, secondly, to measure the impact of self-training on the character detection performance.

### *6.1. Setup*

The following datasets are used for experimental evaluation (see Section 3):

- The **training set** consists of 30,000 synthetically generated pages.
- The **alignment set** includes 899 manuscript pages together with their transcription.
- The independent **test set** contains the same 47 manuscript pages used by Nguyen et al. [3], which were taken from ten different manuscripts.

Please note that because the proposed method is annotation-free, we did not have access to a human-annotated validation set for fine-tuning hyper-parameters of the system. Instead, reasonable defaults have been chosen and have been visually validated on several pages of the alignment set. The test set has only been used to measure the final system performance.

All 899 pages of the alignment set are used for alignment and all 47 pages of the test set are used for evaluating character detection, both before and after self-training. We distinguish two states of the character detection system, accordingly:

- **Synthetic character detection.** The YOLO-based character detection system is called *synthetic* before self-training, because it is trained on synthetic page images.
- **Adapted character detection.** After self-training on the aligned pages it is called *adapted*, because it has been adapted to real manuscript pages. The system is retrained after aligning all pages of the alignment set.

For page synthetization, we consider a total of 30,000 pages with variable scaling of the main text area, number of columns and rows, and image distortions (see Section 5.1). We have also conducted preliminary experiments with 10,000 and 60,000 pages, respectively. While the results for the former were clearly worse, only slight improvements were observed for the latter. Four variants of synthetic pages are considered with respect to the characters used:

- **Printed (1 class).** All 26,969 characters of the Nom font are used for synthetization with uniform distribution. All bounding boxes are labeled with the same class, i.e., character detection is trained for bounding box regression only.
- **Printed (4855 classes)**. Only the 4855 characters actually appearing in the 899 transcribed pages are used for synthetization. Furthermore, the bounding boxes are labeled with the actual character class, i.e., character detection is trained conjointly for bounding box regression and character classification.
- **Handwritten (CASIA)**. Images of traditional Chinese characters from the CASIA-AHCDB [9] are used for synthetization, more specifically all 832,939 images from the training set of the basic category of the first style, including a total of 2365 character classes. The bounding boxes are labeled with one class.
- **Handwritten (Nom)**. Images of Nom characters from the alignment result are used. The bounding boxes are labeled with one class. This scenario is also a form of self-training. However, the synthetic pages that are generated do not include the manuscript background.

Please note that only the characters are different for the four variants. Otherwise, the 30,000 synthetic training pages are generated in the same way.

For YOLO-based object detection, we use a standard implementation from Ultralytics (https://github.com/ultralytics/yolov5; accessed on 25 May 2021) and distinguish the following types of models:

- **YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x.** Different model sizes ranging from 7.3M weights (YOLOv5s) to 87.7M weights (YOLOv5x), pre-trained on the COCO dataset [36] and using hyper-parameters suggested for fine-tuning, ("hyp.finetune.yaml",

version Oct 11, 2020, commit cc03c1d5727e178438e9f0ce0450fa6bdbbe1ea7.) including an initial learning rate of 0.0032.

- **YOLOv5s-scr, YOLOv5m-scr, YOLOv5l-scr, YOLOv5x-scr.** The same model architectures but with random weights and hyper-parameters suggested for training from scratch,("hyp.scratch.yaml", v ersion Nov 17, 2020, commit 41517d9d7328f7e15527a594df342b38294c000a.) including an initial learning rate of 0.01.

Training 25 epochs on 30,000 synthetic pages with a batch size of 24 and two Titan RTX cards takes a few hours for all model sizes.

For clustering-based alignment (see Section 5.2), the meta parameters of our algorithm are set to the suggested default values $\sigma_{size} = 0.2$, $\sigma_{overlap} = 0.1$, and $\sigma_{border} = 5$.

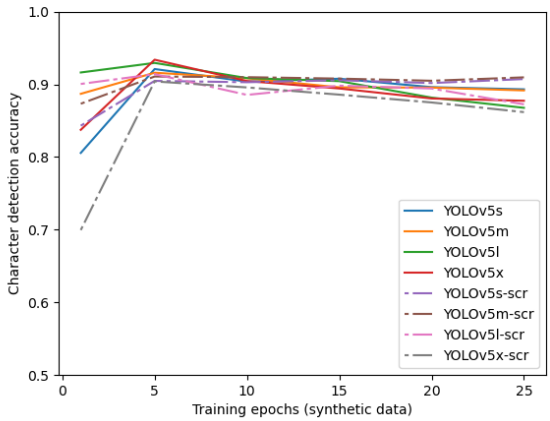
### *6.2. Results*

Table 1 presents the results for *synthetic* character detection on the 47 test pages using the different YOLO models trained 25 epochs on 30,000 synthetic pages. The character detection accuracy per epoch is depicted in Figure 4a. The results show a surprisingly high *synthetic character detection accuracy of about* 90% across all models despite the fact that during training of the object detection method, the models have neither seen handwritten characters nor real page backgrounds. However, Figure 4a also shows an overfitting to the synthetic data after 5 epochs. Besides the IoU, we also indicate the results $IoU_0$ without shrinking the detected bounding boxes. The tight bounding boxes are clearly closer to the ground truth than the raw detection results. Furthermore, the results in Table 2 show a tendency that training from scratch leads to slightly less accurate bounding boxes, i.e., a lower IoU.

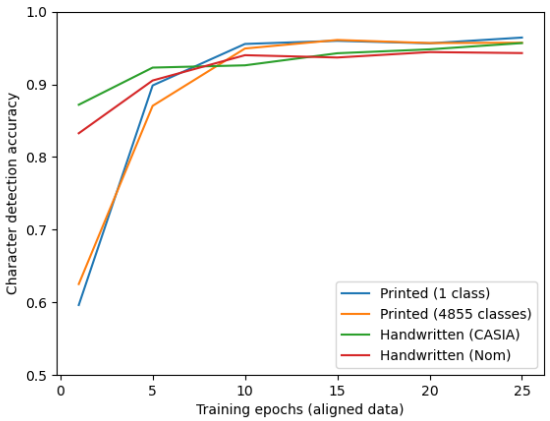
Among the different evaluation measures, the IoU is the most difficult for achieving a perfect score, as the detected bounding box has to fit the ground truth bounding box with pixel-accuracy, something that even different human annotators are not expected to achieve. On the other hand, precision and recall indicate very high scores despite the fact that a visual inspection of the results, for example in Figure 4c, clearly shows deletion and insertion errors. We argue that the character detection accuracy is better suited for evaluating the detection results in a balanced way, including deletions, insertions, and the quality of the bounding box. Boxes with an IoU larger than 0.5 are typically capturing enough detail of a character, such that a subsequent character detection can be successful. Therefore, the correlation between the character detection accuracy and the character recognition accuracy is expected to be high.

Regarding transcription alignment, the YOLOv5s model was able to align 872 pages (97.0%), the YOLOv5m model 875 pages (97.3%), the YOLOv5l model 861 pages (95.8%), and the YOLOv5x model 858 pages (95.4%), i.e., for almost all pages a median column and a median row could be found, allowing building a complete grid for aligning all characters of the transcription. This set of aligned pages is then used for self-training the object detection system.

Table 3 and Figure 4b present the evaluation results for *adapted* character detection after self-training 25 epochs on the aligned pages. The results show a significant improvement when compared with synthetic character detection, ranging from +5.27% for YOLOv5x to +7.27% for YOLOv5m, as shown in Table 4. The results in Table 3 also reaffirm the tendency that training from scratch leads to less accurate bounding boxes in terms of IoU when compared with COCO pre-trained models. Interestingly, changing the training data from synthetic data to real manuscript data leads to a loss in recognition accuracy during the first few epochs, as observed in Figure 4b. The effect is weaker when using handwritten characters for page synthetization. However, after a few epochs, self-training leads to significantly better results when compared with the synthetic system and a plateau is reached, showing no overfitting effect. Contrary to our expectations, reducing the dictionary to the 4855 characters actually present in the 899 alignment pages and training YOLO conjointly for regression and classification did not lead to better results. All synthetization settings eventually lead to an *adapted character detection accuracy of about* 95%.




(**a**) Synthetic character detection. Each epoch includes training on 30'000 synthetic pages generated with printed characters. Comparison of different YOLO models, either COCO-pretrained or trained from scratch (-scr)




(**b**) Adapted character detection with YOLOv5m. Each epoch includes training on about 875 aligned pages. Comparison of printed and handwritten characters used for training the synthetic detection system, for which training is then continued on the aligned pages

(**c**) Adapted character detection with YOLOv5m. Matching bounding boxes $\mathcal{M}$ are green, insertions $\mathcal{I}$ are magenta, deletions $\mathcal{D}$ are red, and the ground truth is blue

**Figure 4.** Evaluation of the character detection performance on the test set.

**Table 1.** Synthetic character detection results on the test set. The performance is measured in terms of character detection accuracy (Acc), precision (P), recall (R), $F_1$ score (F1), and mean intersection over union before ($IoU_0$) and after (IoU) shrinking the detected bounding boxes.

| | Acc | P | R | F1 | $IoU_0$ | IoU |
|---|---|---|---|---|---|---|
| YOLOv5s | 89.31 | 94.45 | 92.20 | 93.12 | 73.76 | 79.63 |
| YOLOv5m | 89.16 | 93.92 | 93.16 | 93.38 | 73.07 | 79.32 |
| YOLOv5l | 86.77 | 92.67 | 91.22 | 91.72 | 71.15 | 78.05 |
| YOLOv5x | 87.74 | 93.81 | 90.58 | 91.96 | 72.19 | 79.32 |
| YOLOv5s-scr | 90.73 | 94.99 | 93.61 | 94.17 | 69.16 | 77.61 |
| YOLOv5m-scr | 90.96 | 95.51 | 93.50 | 94.40 | 69.59 | 77.96 |
| YOLOv5l-scr | 87.25 | 95.41 | 88.24 | 91.45 | 68.39 | 76.57 |
| YOLOv5x-scr | 86.18 | 94.32 | 87.57 | 90.54 | 68.32 | 75.64 |

**Table 2.** Adapted character detection results on the test set.

| | Acc | P | R | F1 | $IoU_0$ | IoU |
|---|---|---|---|---|---|---|
| YOLOv5s | 94.73 | 98.07 | 96.47 | 96.95 | 78.19 | 83.18 |
| YOLOv5m | 96.43 | 98.64 | 97.11 | 97.85 | 78.83 | 85.13 |
| YOLOv5l | 93.51 | 98.09 | 93.91 | 95.85 | 75.61 | 83.93 |
| YOLOv5x | 93.01 | 97.85 | 93.47 | 95.42 | 75.67 | 83.77 |
| YOLOv5s-scr | 96.53 | 97.89 | 97.98 | 97.93 | 73.73 | 77.56 |
| YOLOv5m-scr | 94.34 | 98.09 | 95.37 | 96.67 | 75.77 | 79.08 |
| YOLOv5l-scr | 93.81 | 98.43 | 95.59 | 96.39 | 75.25 | 77.68 |
| YOLOv5x-scr | 88.82 | 94.69 | 95.30 | 93.84 | 71.07 | 74.68 |

**Table 3.** Adapted character detection results with YOLOv5m on the test set. Comparison of printed and handwritten page synthetization (see Figure 3b).

| | Acc | P | R | F1 | IoU |
|---|---|---|---|---|---|
| Printed (1 class) | 96.43 | 98.64 | 97.11 | 97.85 | 85.13 |
| Printed (4855 classes) | 95.72 | 98.31 | 96.70 | 97.47 | 85.26 |
| Handwritten (CASIA) | 95.68 | 98.26 | 96.49 | 97.35 | 85.54 |
| Handwritten (Nom) | 94.30 | 97.94 | 94.96 | 96.38 | 85.20 |

**Table 4.** Impact of annotation-free transcription alignment on the character detection accuracy.

| | Synthetic | Adapted | Improvement |
|---|---|---|---|
| YOLOv5s | 89.31 | 94.73 | +5.42 |
| YOLOv5m | 89.16 | 96.43 | +7.27 |
| YOLOv5l | 86.77 | 93.51 | +6.74 |
| YOLOv5x | 87.74 | 93.01 | +5.27 |

The overall best result is achieved with the COCO pre-trained YOLOv5m model, resulting in the *best character detection accuracy of* 96.43% *and IoU of* 85.13%. We compare the model with the results achieved by Nguyen et al. [3] on the same test images ( Results are taken from Table 2 of [3] with respect to rectangular regions) using U-Net-based semantic segmentation, trained on synthetic data and fine-tuned on human-annotated pages, followed by a watershed algorithm to separate the individual characters. Two of the U-Net backbones are outperformed (VGG-16 and ResNet-50), while the Inception-ResNet-V2 backbone achieves higher results than our method. Table 5 shows the detailed results. Overall, we observe that comparable results are achieved with the proposed alignment method, despite the fact that 10 pages of the test set were used to fine-tune the system in [3], while no fine-tuning to the test manuscripts has been performed for our system. This is a promising outcome for the proposed approach since no human annotations were required to achieve a good generalization to unseen data.

**Table 5.** Comparison with the state of the art.

| | **IoU** |
|---|---|
| U-Net (VGG-16) [3] | 79.67 |
| U-Net (ResNet-50) [3] | 83.17 |
| U-Net (Inception-ResNet-V2) [3] | 90.08 |
| YOLOv5m (ours) | 85.13 |

## 7. Conclusions

The experimental results clearly demonstrate the high potential of object detection methods for the task of transcription alignment. For the historical Vietnamese dataset, such an alignment could be established without asking humans to manually annotate training images, which is a time-consuming and tedious work hampering the progress for digitizing large quantities of historical manuscripts. At the same time, the proposed method is still learning-based, able to adapt to a variety of page layouts, page backgrounds, and handwriting styles by means of self-training.

The high structural similarity between the printed font and the handwriting styles observed in the historical Vietnamese dataset is most likely a key factor for the success of the method. In the future, we aim to investigate whether or not similar segmentation-free, annotation-free, and learning-based approaches are also feasible for other scripts and languages, for example in the context of medieval Latin manuscripts. Furthermore, we are interested in applying similar alignment algorithms to historical Vietnamese steles [37], where the stone background and the character engravings show even more variability when compared with manuscripts.

Finally, a promising line of future research is to attempt an annotation-free transcription using fully convolutional models that are trained without human supervision on scanned manuscript images. This line of research raises the question to what extent human annotations are actually needed for accurate document analysis and recognition, and to what extent printed fonts may already be sufficient as a starting point for self-adaptation to different handwriting styles.

**Author Contributions:** Conceptualization, A.S.-B., M.J., B.W., A.F. and M.B.; methodology, A.S.-B., M.J., B.W., A.F. and M.B.; software, A.S.-B., M.J., B.W. and A.F.; validation, A.S.-B., M.J., B.W. and A.F.; formal analysis, A.S.-B., M.J., B.W. and A.F.; investigation, A.S.-B., M.J., B.W. and A.F.; resources, A.S.-B., M.J., B.W., A.F. and M.B.; data curation, A.S.-B. and A.F.; writing—original draft preparation, A.S.-B., M.J. and A.F.; writing—review and editing, A.S.-B., M.J., B.W., A.F. and M.B.; visualization, A.S.-B., M.J. and A.F.; supervision, A.F. and M.B.; project administration, A.F.; funding acquisition, A.F. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work has been supported by the Swiss Hasler Foundation (project 20008).

**Acknowledgments:** We would like to thank Lee Collins and John Balaban from the Vietnamese Nom Preservation Foundation for providing us with the images and transcriptions of the *Tale of Kieu* and the *Luc Van Tien*. Furthermore, we thank Kha Cong Nguyen, Cuong Tuan Nguyen, and Masaki Nakagawa, the authors of [3], for providing us with their ground truth for the test images.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Fischer, A.; Liwicki, M.; Ingold, R. *Handwritten Historical Document Analysis, Recognition, and Retrieval—State of the Art and Future Trends*; World Scientific: Singapore, 2020.
2. Kornfield, E.M.; Manmatha, R.; Allan, J. Text Alignment with Handwritten Documents. In Proceedings of the International Workshop on Document Image Analysis for Libraries, Palo Alto, CA, USA, 23–24 January 2004; pp. 195–209.
3. Nguyen, K.C.; Nguyen, C.T.; Nakagawa, M. Nom document digitalization by deep convolution neural networks. *Pattern Recognit. Lett.* **2020**, *133*, 8–16.
4. Yang, H.; Jin, L.; Huang, W.; Yang, Z.; Lai, S.; Sun, J. Dense and Tight Detection of Chinese Characters in Historical Documents: Datasets and a Recognition Guided Detector. *IEEE Access* **2018**, *6*, 30174–30183. [CrossRef]

5. Fischer, A.; Keller, A.; Frinken, V.; Bunke, H. Lexicon-Free Handwritten Word Spotting Using Character HMMs. *Pattern Recognit. Lett.* **2012**, *33*, 934–942. [CrossRef]
6. Sarkar, R.; Das, N.; Basu, S.; Kundu, M.; Nasipuri, M.; Basu, D.K. CMATERdb1: A database of unconstrained handwritten Bangla and Bangla–English mixed script document image. *Int. J. Doc. Anal. Recognit.* **2012**, *15*, 71–83. [CrossRef]
7. Romero, V.; Fornés, A.; Serrano, N.; Sánchez, J.A.; Toselli, A.H.; Frinken, V.; Vidal, E.; Lladós, J. The ESPOSALLES Database: An Ancient Marriage License Corpus for Off-line Handwriting Recognition. *Pattern Recognit.* **2013**, *46*, 1658–1669. [CrossRef]
8. Kassis, M.; Abdalhaleem, A.; Droby, A.; Alaasam, R.; El-Sana, J. VML-HD: The historical Arabic documents dataset for recognition systems. In Proceedings of the 1st International Workshop on Arabic Script Analysis and Recognition (ASAR), Nancy, France, 3–5 April 2017; pp. 11–14. [CrossRef]
9. Xu, Y.; Yin, F.; Wang, D.H.; Zhang, X.Y.; Zhang, Z.; Liu, C.L. CASIA-AHCDB: A large-scale Chinese ancient handwritten characters database. In Proceedings of the 15th International Conference on Document Analysis and Recognition (ICDAR), Sydney, Australia, 20–25 September 2019; pp. 793–798. [CrossRef]
10. Kusetogullari, H.; Yavariabdi, A.; Cheddad, A.; Grahn, H.; Hall, J. ARDIS: A Swedish historical handwritten digit dataset. *Neural Comput. Appl.* **2020**, *32*, 16505–16518. [CrossRef]
11. Stamatopoulos, N.; Gatos, B.; Louloudis, G.; Pal, U.; Alaei, A. ICDAR 2013 Handwriting Segmentation Contest. In Proceedings of the 13th International Conference on Document Analysis and Recognition (ICDAR), Washington, DC, USA, 22–28 August 2013; pp. 1402–1406. [CrossRef]
12. Pratikakis, I.; Zagoris, K.; Gatos, B.; Puigcerver, J.; Toselli, A.H.; Vidal, E. ICFHR2016 Handwritten Keyword Spotting Competition (H-KWS 2016). In Proceedings of the 15th International Conference on Frontiers in Handwriting Recognition (ICFHR), Shenzhen, China, 23–26 October 2016; pp. 613–618. [CrossRef]
13. Sánchez, J.A.; Romero, V.; Toselli, A.H.; Villegas, M.; Vidal, E. ICDAR2017 Competition on Handwritten Text Recognition on the READ Dataset. In Proceedings of the 14th International Conference on Document Analysis and Recognition (ICDAR), Kyoto, Japan, 9–15 November 2017. [CrossRef]
14. Bluche, T.; Louradour, J.; Messina, R. Scan, Attend and Read: End-to-End Handwritten Paragraph Recognition with MDLSTM Attention. In Proceedings of the 14th International Conference on Document Analysis and Recognition (ICDAR), Kyoto, Japan, 9–15 November 2017; pp. 1050–1055. [CrossRef]
15. Poulos, J.; Valle, R. Character-based handwritten text transcription with attention networks. *Neural Comput. Appl.* **2021**. [CrossRef]
16. Marti, U.V.; Bunke, H. The IAM-Database: An English Sentence Database for Off-line Handwriting Recognition. *Int. J. Doc. Anal. Recognit.* **2002**, *5*, 39–46. [CrossRef]
17. Leydier, Y.; Eglin, V.; Brès, S.; Stutzmann, D. Learning-free text-image alignment for medieval manuscripts. In Proceedings of the 14th International Conference on Frontiers in Handwriting Recognition (ICFHR), Crete Island, Greece, 1–4 September 2014; pp. 363–368. [CrossRef]
18. Rabaev, I.; Cohen, R.; El-Sana, J.; Kedem, K. Aligning transcript of historical documents using dynamic programming. In Proceedings of the International Conference on Document Recognition and Retrieval (DRR), Catania, Italy, 26–29 October 2015. [CrossRef]
19. Fischer, A.; Frinken, V.; Fornés, A.; Bunke, H. Transcription alignment of Latin manuscripts using hidden Markov models. In Proceedings of the 1st International Workshop on Historical Document Imaging and Processing (HIP), New York, NY, USA, 18 May 2011; pp. 29–36.
20. Romero-Gómez, V.; Toselli, A.H.; Bosch, V.; Sánchez, J.A.; Vidal, E. Automatic Alignment of Handwritten Images and Transcripts for Training Handwritten Text Recognition Systems. In Proceedings of the Workshop on Document Analysis Systems (DAS), Vienna, Austria, 24–27 April 2018; pp. 328–333.
21. Chammas, E.; Mokbel, C.; Likforman-Sulem, L. Handwriting Recognition of Historical Documents with Few Labeled Data. In Proceedings of the Workshop on Document Analysis Systems (DAS), Vienna, Austria, 24–27 April 2018; pp. 43–48. [CrossRef]
22. Ziran, Z.; Pic, X.; Innocenti, S.U.; Mugnai, D.; Marinai, S. Text alignment in early printed books combining deep learning and dynamic programming. *Pattern Recognit. Lett.* **2020**, *133*, 109–115.
23. Yin, F.; Wang, Q.F.; Liu, C.L. Transcript mapping for handwritten Chinese documents by integrating character recognition model and geometric context. *Pattern Recognit.* **2013**, *46*, 2807–2818.
24. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
25. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.E.; Fu, C.Y.; Berg, A. SSD: Single Shot MultiBox Detector. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 11–14 October 2016; pp. 21–37.
26. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal loss for dense object detection. In Proceedings of the International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2980–2988. [CrossRef]
27. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6517–6525. [CrossRef]
28. Ren, S.; He, K.; Girshick, R.B.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *39*, 1137–1149.
29. Farhadi, A.; Redmon, J. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.

30. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
31. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
32. Wang, C.Y.; Liao, H.; Yeh, I.H.; Wu, Y.H.; Chen, P.Y.; Hsieh, J.W. CSPNet: A New Backbone that can Enhance Learning Capability of CNN. In Proceedings of the International Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 14–19 June 2020; pp. 1571–1580.
33. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path aggregation network for instance segmentation. In Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768. [CrossRef]
34. Jocher, G. Ultralytics/Yolov5: v4.0—nn.SiLU() Activations, Weights & Biases Logging, PyTorch Hub Integration. 2021. Available online: https://doi.org/10.5281/ZENODO.4418161 (accessed on 25 May 2021). [CrossRef]
35. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS), Vancouver, BC, Canada, 16–23 May 2019.
36. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. In Proceedings of the 13th European Conference on Computer Vision (ECCV), Zurich, Switzerland, 6–12 September 2014; pp. 740–755.
37. Scius-Bertrand, A.; Voegtlin, L.; Alberti, M.; Fischer, A.; Bui, M. Layout Analysis and Text Column Segmentation for Historical Vietnamese Steles. In Proceedings of the 5th International Workshop on Historical Document Imaging and Processing (HIP), Sydney, NSW, Australia, 20–21 September 2019; pp. 84–89.