Show, don't tell –

Providing Visual Error Feedback for Handwritten Documents

Said Yasin[1], Torsten Zesch[1]

[1]CATALPA - Center of Advanced Technology for Assisted Learning and Predictive Analytics
said.yasin@studium.fernuni-hagen.de
torsten.zesch@fernuni-hagen.de

Abstract

Handwriting remains an essential skill, particularly in education. Therefore, providing visual feedback on handwritten documents is an important but understudied area in NLP. We outline the many challenges when going from an image of handwritten input to correctly placed informative error feedback. We empirically compare modular and end-to-end systems and find that both approaches currently do not achieve acceptable overall quality. We identify the major challenges and outline an agenda for future research.

Keywords: HWR, Error feedback, Modular vs end-to-end

1. Motivation

Even in the digital age, handwriting is still the primary writing method used in schools worldwide (Freedman et al., 2016). It remains an important skill e.g. in early language acquisition (Ray et al., 2022), writing development (Feng et al., 2019) or for note taking (Mueller and Oppenheimer, 2014). Therefore, the ability to provide digital feedback on handwritten work is essential for bridging the gap between physical documents and current NLP technology. The bottom image in Figure 1 shows an example of such feedback. However, significant challenges remain in word detection, recognition, ordering, and feedback generation, all of which are crucial for providing feedback in handwriting.

Traditional handwriting recognition (HWR) typically tackles these steps in a modular fashion, where detection, ordering, and recognition are treated as independent, sequential steps (Coquenet, 2022). While this approach allows for flexible optimization of state-of-the-art text components, it also leads to error accumulation as mistakes propagate between steps (Liu et al., 2022). Figure 2 illustrates this by showing how erroneous detection influences recognition. If a bounding box (BB) is incorrectly positioned, recognition is likely to fail as the correct word cannot be extracted from a truncated BB. Despite this limitation, modular systems continue to dominate HWR (Coquenet, 2022). Moreover, previous works on feedback systems (Karlsson and Åkerman, 2022; Imasha et al., 2023) insufficiently analyze the challenges in detection, ordering, and recognition.

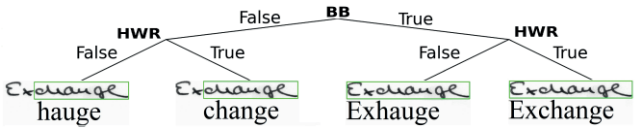


Fig. 2: Four scenarios in modular word detection and recognition impacting feedback accuracy

To bridge this gap, we provide a systematic analysis of each necessary step and its associated challenges in mapping error feedback onto handwritten text, along with an empirical comparison of modular and end-to-end systems. We also develop a prototype using a document camera. We make the source code for the prototype publicly available to foster future research in that area.[1]

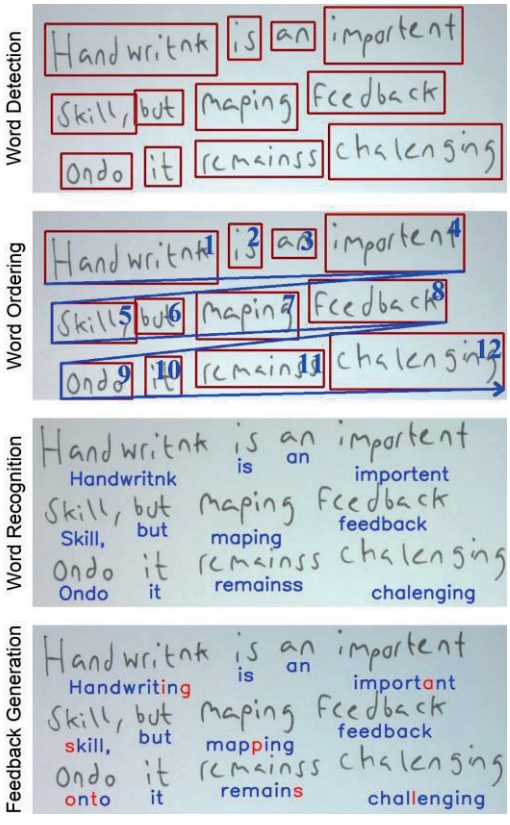


Fig. 1: Key processing steps in providing visual error feedback for handwriting

[1] https://github.com/said702/Prototyp_HWR_Feedback

Said Yasin, Torsten Zesch, *Show, Don't Tell – Providing Visual Error Feedback for Handwritten Documents*. In: Z. Vetulani, P. Paroubek, I. Kardava (Eds.), Human Language Technologies as a Challenge for Computer Science and Linguistics – 2025, pp. 99–103. Poznań: Adam Mickiewicz University Press, 2025. © Authors, 2025. https://doi.org/10.14746/amup.9788323245414.18. Open Access chapter, distributed under the terms of the CC licence (BY-NC-ND, https://creativecommons.org/licenses/by-nc-nd/4.0/).

2. Feedback on Handwritten Input

As we have discussed in the introduction, providing visual feedback on handwritten documents requires a series of steps to be performed (word detection, word ordering, word recognition, and feedback generation) that all come with their own challenges. In this section, we now discuss each step in more detail.

2.1. Word Detection

Accurate word detection, i.e. putting a well-fitting BB around a word in an image, is the essential first step for providing feedback on handwritten documents (see top image in Figure 1). Word detection is evaluated using the Intersection over Union (IoU) metric, which quantifies the overlap between two BBs by computing the ratio of their area of intersection to the area of their union. IoU values may range from 0 to 1, where a higher value indicates greater overlap (Lucas et al., 2005). Using a threshold θ, which defines the minimum overlap between detected and correct BBs, a system's accuracy can be evaluated. To illustrate the impact of different thresholds on feedback placement, we plotted 100 random BBs for IoUs between 0.5 and 1 on an image snippet from the IAM dataset (see Figure 3). With decreasing IoU thresholds, BBs deviate markedly from their original positions. The deviation is more pronounced horizontally due to the rectangular shape of BBs. In light of these results, the commonly used IoU threshold of 0.5 for word detection should be reconsidered for our use case, as it may lead to inaccurate feedback placement. However, maintaining this threshold ensures comparability with other studies (Abdo et al., 2024; Karlsson and Åkerman, 2022), but it may lead to an overestimation of accuracy. For example, Karlsson and Åkerman (2022) report a detection accuracy of over 95 % based on an IoU threshold of 0.5. In contrast, Abdo et al. (2024) show that while similarly high accuracy can be achieved at this threshold, performance drops significantly when stricter thresholds up to 0.95 are applied.

2.2. Word Ordering

Once words are detected, the next step is to correctly order them to preserve context. In modular approaches, text ordering is determined independently of detection and extraction, while in end-to-end systems, it is learned in conjunction with these processes. In the feedback system by Karlsson and Åkerman (2022), word ordering was achieved using a heuristic algorithm that first calculates the average box height, groups boxes based on their proximity along the y-axis, and then sorts them by their x-coordinate. In contrast, Coquenet (2022) introduced an end-to-end system that directly learns text ordering using a Vertical Attention Network, demonstrating that it outperforms a traditional modular approach. Rule-based ordering becomes tedious for complex documents and requires prior knowledge of reading order, such as whether it is column-based or row-based (Coquenet, 2022). Additionally, left-to-right heuristics struggle with skewed handwritten text. Word ordering can be evaluated using the Normalized Spearman's Footrule Distance (NSFD), which measures the normalized sum of absolute positional differences (Vidal et al., 2023), or indirectly via the Bilingual Evaluation Understudy (BLEU) score, which compares n-grams between predicted and reference text (Birch et al., 2010).

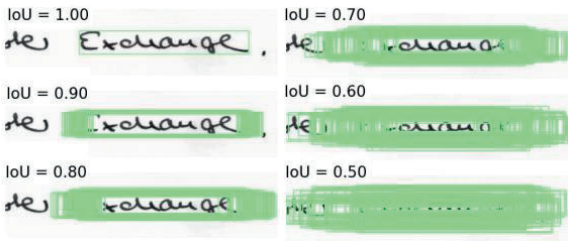


Fig. 3: Placement of 100 random BBs with specific IoUs

2.3. Word Recognition

In this step, the handwritten word within the BB is sent to a recognition system that returns the recognized text string. Performance is usually measured by Character Error Rate (CER), which quantifies the number of incorrectly recognized characters (Neudecker et al., 2021). Karlsson and Åkerman (2022) review 31 studies on HWR systems using the IAM dataset, reporting CER values ranging from 2.9% to 17%, with the TrOCR model (Li et al., 2021) performing best. In light of these results, current feedback systems for real-time spelling error detection demonstrated competitive CERs, with Karlsson and Åkerman (2022) achieving 3.4% using TrOCR on IAM test data and Imasha et al. (2023) reporting 3.4% on the IAM and SROIE datasets through a Transformer-based extraction approach. In comparison, the end-to-end HWR system based on a Vertical Attention Network (Coquenet, 2022) capable of processing entire documents while simultaneously recognizing both text and layout, achieved a CER of 3.6% on the READ 2016 dataset. While low single digit CER values seem quite good, this performance level might still severely impact error feedback. For example, a CER of 5% means that every 20th character is incorrectly recognized, which might lead to spurious errors being flagged. However, CER values alone do not tell the whole picture. For example, Gold et al. (2023) point out that HWR systems either under-correct by misinterpreting characters (e.g. reading *d* as *cl*) or over-correct by normalizing misspellings, which erases learner errors and hinders feedback.

2.4. Feedback Generation

In this study, we focus on error feedback, which involves identifying and marking mistakes in the extracted text, such as spelling and grammatical errors, as well as segmentation errors like split compounds and merged words (Stymne et al., 2013). While spelling and segmentation errors are easier to highlight because they affect only individual words or characters, grammar mistakes are more challenging as they often would require showing structural modifications. A major challenge for visual feedback is placement, i.e. we need to position the feedback in physical space in a way that minimizes distractions. For example, if there is not enough space between two handwritten lines, the feedback might overlap with the handwriting leading to readability issues, as shown in Figure 4.

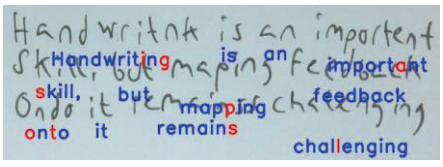


Fig. 4: Feedback placement with insufficient space

One possible solution, inspired by language learning support (Rzayev et al., 2020), is to overlay feedback directly on the text. However, this hides the original text making it hard for learners to understand the nature of their mistake. Another approach from language learning support (Mosa and Katsuhiko, 2014) is to color-highlight individual letters of the original word, enabling detailed annotation of errors. However, this would require locating single characters within a word, while detection approaches rely on word-level BBs. Also, most handwritten datasets are annotated at the word or line level (Marti and Bunke, 2002; Krishnan et al., 2021) making it challenging to handle character-level feedback placement. However, note that (as is shown in the bottom image in Figure 1) we can still provide character-level feedback in form of highlighting the misspelled characters in the word. What we currently cannot do is directly showing the feedback at the correct character position.

3. Experimental Setup

To address the challenges of providing feedback on handwritten documents, as discussed in the previous section, we empirically examine the effectiveness of modular and end-to-end approaches.

3.1. Data

We use two datasets: IAM and Imgur5K.

The IAM dataset (Marti and Bunke, 2002) is derived from the Lancaster-Oslo-Bergen Corpus and consists of 1,539 scanned handwritten English forms, penned by 657 different writers, with a total of 13,353 isolated text lines and 115,320 words. The dataset includes metadata stored in an XML file containing the BB positions and the transcriptions of the handwritten words, which are arranged according to the reading direction, while the entire text is also displayed as printed text in the header of each image.

The Imgur5K dataset (Krishnan et al., 2021) comprises 8,177 image pages containing 230,570 words from a diverse range of real-world handwritten samples by nearly 5,300 writers, originally sourced from public posts on Imgur.com. These samples include items such as notes, logos, and various other objects, unlike the form-based samples in the IAM dataset, making Imgur5K particularly challenging. The dataset also includes metadata in the form of a JSON file, which contains BB positions, transcribed words, and angle information because of variable text orientation. However, since the words are annotated at the word level and arranged in various layouts, the metadata does not contain explicit ordering information.

Preprocessing All images are resized to 800×800 pixels, as our experiment showed that this leads to improved results. No further preprocessing is applied, as tests with OpenCV processing methods showed no significant overall benefit.

3.2. Approaches

In this study, we limit ourselves to systems capable of generating BBs, a feature lacking in many systems (Wigington, 2023) and crucial for feedback placement.

End-to-end system We selected PGNet (Wang et al., 2021), an end-to-end OCR system from PaddleOCR, due to the limited availability of research and open-source end-to-end HWR systems (Coquenet, 2022), as well as its efficiency in avoiding complex, time-consuming steps like non-maximum suppression and region of interest. The basic idea of PGNet is to combine text detection and recognition by sharing CNN features and training both components simultaneously. The system learns text ordering by organizing center points via directional offsets, enabling recognition in non-traditional reading directions.

Modular systems We use Tesseract and EasyOCR, Tesseract detects text using a binarization algorithm, followed by Convolutional Neural Networks (CNNs) for character recognition and Recurrent Neural Networks with Long Short-Term Memory for sequence processing (Thammarak et al., 2022). EasyOCR uses Character Region Awareness for Text Detection (Baek et al., 2019), which relies on CNNs, and utilizes a Convolutional Recurrent Neural Network (Shi et al., 2015) for extraction.

Cloud For comparison, we also use the Handprint library through Google Cloud Vision API, whose specific methods remain undisclosed (Thammarak et al., 2022).

3.3. Evaluation

We evaluate **word detection** using precision, recall, and F1 score, considering a BB correctly placed if IoU ≥ 0.5 to ensure comparability with related work. To account for the limitations of this threshold (see Section 2.1), we also plot the distribution of all IoU values. We evaluate **word recognition** using the CER. Only successfully detected words are considered, ensuring that undetected words do not contribute a CER of 1.0. We evaluate **word ordering** using the BLEU score and NSFD metrics. To exclude recognition errors, we use the system's proposed order and replace predicted words with the ground truth.

4. Empirical Analysis

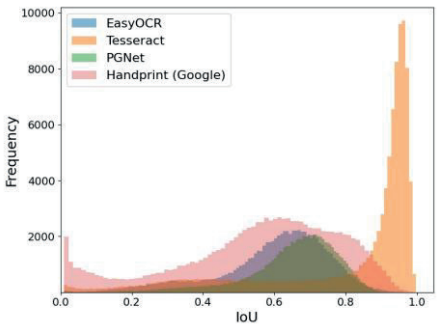
This section analyzes the performance of modular and end-to-end systems across different datasets, highlighting their strengths and weaknesses while providing valuable insights for developing an effective feedback system.

4.1. Word Detection

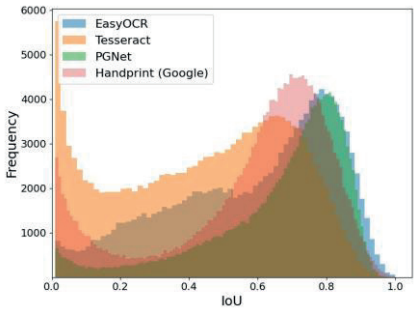
Table 1 shows the results based on an IoU threshold of 0.5, while Figure 5 illustrates the distribution of all calculated IoUs. The end-to-end (E2E) system PGNet achieve good overall performance, while Tesseract performs well on IAM but poorly on Imgur5k. However, the IoU histogram for IAM (see Figure 5a) reveals unexpectedly high IoUs for Tesseract, possibly indicating data contamination due to undocumented training on the IAM dataset. Notably, Handprint achieved nearly identical accuracy on both datasets. The IoU histogram of Imgur5k (see Figure 5b) shows that PGNet and the Handprint library achieve the highest IoUs with similar distributions.

Type	Model	IAM			Imgur5k		
		P	R	F	P	R	F
E2E	PGNet	.83	.60	.70	.62	.46	.53
Modular	EasyOCR	.74	.58	.65	.58	.41	.48
Modular	Tesseract	.78	.74	.76	.30	.28	.29
Cloud	Handprint	.65	.65	.65	.67	.67	.67

Table 1. Word detection results



(a) IAM



(b) Imgur5K

Fig. 5: IoU histograms

Compared to the YOLOv5l detection system trained on the IAM, which achieved 96.1% recall and 96.2% mAP with the same IoU threshold of 0.5 (Karlsson and Åkerman, 2022), the evaluated systems achieve worse results. However, their model was trained and tested on IAM (in-distribution), while our evaluation includes out-of-distribution scenarios.

4.2. Word Ordering

Since the IAM dataset provides a predefined reading order, we evaluated the models' ordering performance using the NSFD and BLEU metrics (see Table 2). The modular system Tesseract achieves the best ordering score, likely influenced by its higher recall (see Table 1), which allowed more words to be considered in the evaluation. However, this may also be partly due to data contamination as described above.

Type	Model	IAM	
		BLEU ↑	NSFD ↓
E2E	PGNet	.21	.18
Modular	EasyOCR	.23	.24
Modular	Tesseract	.17	.50
Cloud	Handprint	.21	.32

Table 2. Word ordering results

4.3. Word Recognition

Table 3 gives an overview of the recognition results.Our evaluation of extraction performance revealed that the Handprint library with Google service achieved the lowest CERs by a significant margin, which is unsurprising given that the other systems are OCR-specific models. When focusing solely on the OCR systems, E2E PGNet achieved the lowest CERs, although its IoU distribution on IAM (see Figure 5a) closely resembles EasyOCR's, suggesting improved error propagation handling.

Compared to advanced HWR systems, which achieve a word-level CER of 4.9% on IAM and 9.3% on Imgur5k (Kass and Vats, 2022), the Handprint library with Google service achieves slightly better performance.

Type	Model	CER values ↓ IAM	Imgur5k
E2E	PGNet	37	50
Modular	EasyOCR	63	54
Modular	Tesseract	43	54
Cloud	Handprint	3	8

Table 3. Word recognition results

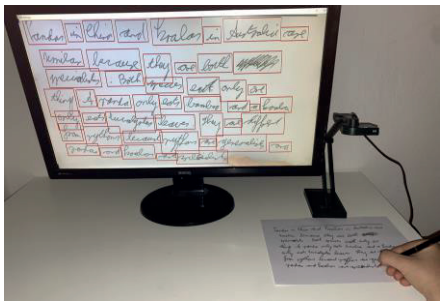
4.4. Summary of Empirical Findings

Our experiments indicate that neither modular nor end-to-end systems provide sufficient quality for visual feedback on handwriting in classroom settings. While some systems excel in specific categories, their overall performance remains inadequate. Achieving high accuracy across detection, ordering, and extraction requires further research on HWR systems, independent of their architectural

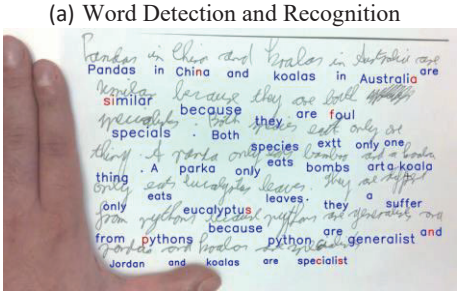
5. Prototype & Future Work

Even if we have concluded from our empirical evaluation that current system performance is probably not ready for classroom use, we still wanted to create a proof of concept where we can experience the feedback quality firsthand. It is well known that HWR results vary considerably with handwriting style (Gold et al., 2021), so for clearly written block letters performance might already be sufficient. We implemented a real-time Python application that uses a document camera to capture handwritten text and display error feedback on the screen. Initial tests confirm reliable BB detection on document camera images, even adapting to sheet movements within milliseconds (see Figure7a). HWR and subsequent error identification using a spellchecker are also fast with GPU acceleration, allowing near real-time feedback. The feedback mapping also works well when there is sufficient space between lines (see Figure 7b). We observed challenges when corrections differed in length from the original word, making it unclear which letters to highlight automatically.

A prototype like this could already support classroom use, for instance by allowing students to have their writing samples automatically checked at a document camera station. However, to avoid misleading learners with inaccurate feedback, it is essential to first address the challenges in word detection, ordering and recognition. Additionally, future work should focus on developing methods for character level BBs detection in handwriting, to enable more precise error annotation on the original text. Further efforts should also consider the development of HWR systems that minimize overcorrection, thereby ensuring that learner errors are preserved. Finally, practical implementations should shift from screen-based setups to augmented reality solutions, such as head-mounted displays, projecting feedback directly onto handwritten text.



(a) Word Detection and Recognition



(b) Feedback Generation

Fig. 7: Prototype setup

References

Abdo, H., A. A. Aisslab, R. Manza, M. Talo, Y. Gu, and S. Bawiskar. 2024. End-to-end deep learning framework for Arabic handwritten legal amount recognition and digital courtesy conversion. *Mathematics*, 12:2256.

Baek, Y., B. Lee, D. Han, S. Yun, and H. Lee. 2019. Character region awareness for text detection.

Bhunia, A. K., S. Khan, H. Cholakkal, R. M. Anwer, F. S. Khan, and M. Shah. 2021. Handwriting transformers.

Birch, A., M. Osborne, and P. Blunsom. 2010. Metrics for MT evaluation: Evaluating reordering. *Mach. Transl.*, 24(1):15–26.

Coquenet, D. 2022. *Towards end-to-end handwritten document recognition*. PhD Thesis.Feng, L., A. Lindner, X. R. Ji, and R. M. Joshi. 2019. The roles of handwriting and keyboarding in writing: A meta-analytic review. *Read. Writ.*, 32(1):33–63.

Freedman, S. W., G. A. Hull, J. M. Higgs, and K. P. Booten. 2016. Handwriting instruction: A commentary on five studies. *Read. Writ.*, 33:20–27.

Gold, C., R. Laarmann-Quante, and T. Zesch. 2023. Recognizing learner handwriting retaining orthographic errors for enabling fine-grained error feedback. In *Proc. BEA 2023*, pages 352–360.

Gold, C., D. van den Boom, and T. Zesch. 2021. Personalizing handwriting recognition systems with limited user-specific samples. In *Proc. ICDAR 2021*, pages 413–428.

Imasha, A., K. Wimalaweera, M. Maddumage, D. Gunasekara, K. Manathunga, and D. Ganegoda. 2023. Pocket English master – Language learning with reinforcement learning, augmented reality and artificial intelligence. In *Learning Technologies and Systems*, pages 74–85.

Krishnan, P., R. Kovvuri, G. Pang, B. Vassilev, and T. Hassner. 2021. TextStyleBrush: Transfer of text aesthetics from a single example. *CoRR*, abs/2106.08385.

Li, M., T. Lv, L. Cui, Y. Lu, D. A. F. Florêncio, C. Zhang, Z. Li, and F. Wei. 2021. TrOCR: Transformer-based optical character recognition with pre-trained models. *CoRR*, abs/2109.10282.

Liu, P., J. Xu, H. Zhou, Z. Liu, W. Zhang, and Y. Yu. 2022. Optimal boxes: Boosting end-to-end scene text recognition by adjusting annotated bounding boxes via reinforcement learning. *arXiv*.

Lucas, S., A. Panaretos, L. Sosa, A. Tang, S. Wong, R. Young, K. Ashida, H. Nagai, M. Okamoto, H. Yamamoto, H. Miyao, J. Zhu, W. Ou, C. Wolf, J.-M. Jolion, L. Todoran, M. Worring, and X. Lin. 2005. ICDAR 2003 robust reading competitions: Entries, results and future directions. *IJDAR*, 7:105–122.

Marti, U.-V., and H. Bunke. 2002. The IAM-database: An English sentence database for offline handwriting recognition. *IJDAR*, 5:39–46.

Mosa, A., and K. Katsuhiko. 2014. A way of supporting non-Arabic speakers in identifying Arabic letters and reading Arabic script in an e-learning system. In *Proc. e-Learning*, pages 123–129.

Mueller, P. A., and D. M. Oppenheimer. 2014. The pen is mightier than the keyboard: Advantages of longhand over laptop note taking. *Psychol. Sci.*, 25(6):1159–1168. PMID: 24760141.

Neudecker, C., K. Baierer, M. Gerber, C. Clausner, A. Antonacopoulos, and S. Pletschacher. 2021. A survey of OCR evaluation tools and metrics. In *Proc. HIP 2021*, pages 13–18.

Karlsson, V., and A. Åkerman. 2022. *Real-time detection of spelling mistakes in handwritten notes*. Master's Thesis.

Kass, D., and E. Vats. 2022. AttentionHTR: Handwritten text recognition based on attention encoder-decoder networks, pages 507–522.

Ray, K., K. Dally, L. Rowlandson, K. I. Tam, and A. E. Lane. 2022. The relationship of handwriting ability and literacy in kindergarten: A systematic review. *Read. Writ.*, 35(5):1119–1155.

Rzayev, R., S. Hartl, V. Wittmann, V. Schwind, and N. Henze. 2020. Effects of position of real-time translation on AR glasses. In *Mensch und Computer*, pp. 251–257.

Shi, B., X. Bai, and C. Yao. 2015. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE TPAMI*, PP.

Stymne, S., E. Pettersson, B. Megyesi, and A. Palmér. 2013. Annotating errors in student texts: First experiences and experiments. Technical report, Uppsala University.

Thammarak, K., P. Kongkla, Y. Sirisathitkul, and S. Intakosum. 2022. Comparative analysis of Tesseract and Google Cloud Vision for Thai vehicle registration certificate. *Int. J. Electr. Comput. Eng.*, 22:1849–1858.

Vidal, E., A. H. Toselli, A. Ríos-Vila, and J. Calvo-Zaragoza. 2023. End-to-end page-level assessment of handwritten text recognition. *Pattern Recognit.*, 142:109695.

Wang, P., C. Zhang, F. Qi, S. Liu, X. Zhang, P. Lyu, J. Han, J. Liu, E. Ding, and G. Shi. 2021. PGNet: Real-time arbitrarily-shaped text spotting with point gathering network. In *Proc. CVPR*.

Wigington, C. 2023. Multi-task CTC for joint handwriting recognition and character bounding box prediction. In *Proc. DocEng 2023*.