Human-in-the-Loop Mass Transcription and Ground Truth Annotation for Challenging Historical Documents

Norbert Fischer, Frank Puppe
Julius-Maximilians-Universität Würzburg
Sanderring 2, 97070 Würzburg, Germany
norbert.fischer@uni-wuerzburg.de, frank.puppe@uni-wuerzburg.de

Abstract

Challenging historical documents still pose significant difficulties for fully automatic layout detection and text recognition, requiring lengthy, demanding correction. We describe our experiences with complex layouts and present our workflow with AdaptOCR, a web-based annotation tool designed to facilitate the efficient transcription and ground-truth annotation of demanding historical documents. Addressing the limitations of existing solutions, AdaptOCR prioritizes a streamlined workflow with an integrated trainable layout and OCR pipeline. The tool uses the PAGE standard to represent document structure and enables the annotation of baselines, regions, text lines and the correction of their transcriptions providing automatic OCR invocation and dictionary-based error detection. Furthermore, it supports flexible annotations with custom element types and attributes to cater to different project requirements. We demonstrate the effectiveness of the workflow and tool in two demanding applications: The transcription of a large corpus of historical printings and the detection / annotation of handwritten artifacts within the private library of the Grimm brothers. In addition, we evaluate the dictionary-based correction and assess the efficiency improvements using AdaptOCR in a pilot study.

Keywords: transcription workflow, annotation software, historical documents

1. Introduction

The transcription and annotation of historical documents play an essential role in both developing historical material and editorial projects within the humanities. In recent years, significant advances have been made in automatic document processing using machine learning. Recently, due to the development of Large Language Models, it has become increasingly easier to render historical archives accessible and to further extract insights. Yet, in many cases, the recognition of complex elements such as handwritten notes is still error-prone, even with modern technologies. Consequently, human review remains essential to the process of cataloging historical works if a high-quality result is desired.

In this paper, we share and generalize our experiences from two challenging mass-transcription projects with complex layouts and careful proofreading and correction of OCR (optical character recognition) and HTR (handwritten text recognition) transcriptions. Our workflow and editor AdaptOCR are similar to well-known tools like Transkribus, OCR4all and eScriptorium (see chapter 3) and use PAGE (Pletschacher and Antonacopoulos, 2010) as underlying representation, but offer additional features that significantly increase overall efficiency. In particular, an own step for the detection of critical elements is supported (e.g., sparse handwritten annotations in books or Greek citations in a Latin text), since they require specialized transcription models and validation procedures. The models for layout recognition and OCR/HTR, including a dictionary, can be fine-tuned and applied during the editing process to improve transcription accuracy continuously. The editor displays the original text line and its transcription directly beneath each other and highlights unknown words using the dictionary to increase correction speed and accuracy.

In the next chapter, the general transcription workflow and the possible annotations are described, and related work is discussed in chapter 3. Next our editor AdaptOCR is described and compared with the related work. In chapter 6, we present two challenging application scenarios. For each scenario, we present an evaluation of the efficiency of the workflow in chapter 7. Finally, we conclude with a summary and future work.

2. Annotation Workflow

The fundamental requirements for a full digital replication of a source document are both layout annotations and a text transcription. The former defines the individual content elements of a digitized page and it's structure, the latter aims to represent the text content digitally. Crucially, the layout annotation, in particular the annotation of text elements, is a prerequisite for the transcription of the text, as it defines what needs to be transcribed. Therefore, machine learning is first applied to detect the layout and manually corrected if necessary. Afterwards, text recognition models are applied to the detected text elements which creates an initial transcription.

A typical machine-learning pipeline for layout recognition consists of text detection using e.g. instance segmentation methods (e.g. Islam et al.,

Proceedings of the Fifteenth Language Resources and Evaluation Conference (LREC 2026), pages 7023–7033
11–16 May 2026.

2023) or baseline detection methods (e.g. Grüning et al., 2019). To detect other layout elements, such as handwritten notes or images, instance segmentation models, such as YOLO (Jocher et al., 2023) can be trained for the requirement element types. Furthermore, semantic segmentation models can be utilized to detect text classes (Xu et al., 2017), or to detect other elements, e.g. graphical elements (Monnier and Aubry, 2020). Detected text lines are then transcribed using an OCR or HTR, such as kraken (Kiessling, 2019), Calamari (Wick et al., 2020), or TrOCR (Li et al., 2023). Optionally, if image quality is insufficient, pre-processing, such as deskewing or dewarping, might be required prior to analyzing the page contents. This processing pipeline is often similar across different applications, although the specific algorithms, used machine learning models, and the degree of automation is different. However, the details of the layout detection can vary greatly: Some applications might utilize just a text detection and ignore detecting other content elements such as images, which then have to be annotated manually if required by the transcription guidelines.

In contrast to building pipelines of several models, there also exist end-2-end approaches. These models, such as DAN (Coquenet et al., 2023), make it possible to directly produce the transcription including structural annotations from the input images in one step. Another approach is the application of multimodal large language models (LLMs), as demonstrated by Humphries et al. (2025) for the transcription of full handwritten pages. Currently, end-2-end approaches provide good results for documents with simple layouts, but have difficulties with complex layout types.

Continuous training and fine-tuning of the used machine learning models during the transcription process is an important part to steadily improve the quality of the automatic detection. In particular, fine-tuning of the OCR models with the newly corrected text during the workflow improves the quality of the text recognition (Reul et al., 2019) and therefore the overall transcription efficiency.

For complex layouts, we give an overview of layout element types and their attributes based on the PAGE standard (see. Tab. 1). The layout contains individual region elements, which form the basic building blocks of the layout annotation, representing text elements (e.g. marginalia, page number, a paragraph of text, etc.) or non-text elements such as images or drawings. Regions might need to be ordered hierarchically (i.e. one region might describe a column which contains child regions describing graphics, handwritten notes and text regions inside the column). Another issue might be documents containing multiple languages or mixed handwritten and printed text. In such cases

PAGE Elements	Required Attributes
Book	production, author
- Page	type, production, author
- -TextRegion	type (header, paragraph, marginalia, etc.), production (handwritten, printed), author, etc.
- - -Text Line	text, language (latin, greek, german, mixed)
- - - -Baseline	
- - ImageRegion	
- - GraphicRegion	Handwritten-annotation-type (underline, strike-through, margin-mark, dot-underline)
- - TableRegion	schema
- - CustomRegion	
- - etc.	

Table 1: Structure of elements and their required attributes for annotation for challenging historical printings following the PAGE standard. Potentially required custom extensions to PAGE are highlighted in red. Child elements are indented relative to their respective parent using dashes.

regions require further annotation with attributes (depending on the transcription guidelines) such as the production type (i.e. handwritten or printed), the language of the text, or the author of a handwritten artifact. Challenging tasks require these annotations even on line-level, e.g. in multilingual documents with heavy language mixing. Such a classification could further impact the subsequent OCR, e.g. when the usage of special models for different languages is necessary.

As different projects pose different goals and challenges, a wide variety of layout annotation features are necessary, although in single projects only some of them are used. To summarize, the existence of different transcription guidelines places high demands on the adaptability of the annotation tool used, as all of these cases must be covered in order to be applicable to different real world applications.

The next step is the correction of the detected text elements. As the initial transcription is provided by machine-learning models, the correction effort depends on the availability of good OCR models for the task. For good transcriptions, spotting the remaining sparse errors can be difficult. When neither sufficiently good pre-trained models nor enough training data is available, corrections demand substantial time. This shifts the focus from the capabilities of the automatic recognition models to the ergonomics and efficiency of the editing application. We present an example for such difficult material in section 6.2.

2.1. PAGE annotation standard

Every workflow requires a suitable knowledge representation. For representing historical document pages and their annotation, several standards exist. The most popular standards are PAGE (Pletschacher and Antonacopoulos, 2010) and ALTO[1]

In this paper we focus on the PAGE standard, as it's commonly used in the community, e.g. the OCR-D project[2], to annotate digitized historical books and documents. PAGE is an XML format encoding the structure and contents of one page. The main building block of an annotation are regions, which describe a part of the page, typically bounded by a polygon or rectangle. Available region types include common elements like text, graphics, and images, plus specialized ones for tables, charts, formulas etc. Regions can further be described with a defined set of attributes based on the type, e.g. "primaryLanguage" for text regions. In many cases, attributes are enumerations limiting the set of possible values. Regions can be organized hierarchically, therefore each region can contain other regions. To annotate the textual contents, a text region can be assigned a transcription itself, or be further split into individual text lines, which can further be split into words and glyphs. PAGE can be extended in different ways using standardized placeholders for custom regions and attributes, providing the flexibility needed for custom workflows and specialized processing pipelines.

2.2. Extensions to PAGE

Some of the required elements (see Tab. 1) can not be directly represented in PAGE. In the following we define these missing annotation elements.

Additional / Custom Attributes Sometimes, more attributes are needed than the ones defined in the PAGE standard. For example, in our application, an author must be assigned to a graphic element or a handwritten text region.

Baselines without text regions or text lines In PAGE, baselines of text can only be stored as a child of a text line element. This conflicts with our workflow, especially when baselines should be annotated for training: In this case, line or region annotations are not required.

Book-Level Attributes PAGE is designed to store information about a single page. However, book-level attributes which are inherited by the pages and elements and only overwritten if necessary allows for a representation of information without repetition.

3. Related Work

Transkribus (Kahle et al., 2017) is a commercial web platform offering an online annotation and correction tool. It integrates a layout analysis and text recognition with pre-trained or custom fine-tuned models, which can be shared on the platform with other users. The editor enables annotation of regions, text lines and baselines. Individual layout elements can be tagged using custom types. Transkribus enables the training of "field models" to detect such tagged elements. Furthermore, the text editor supports the annotation of named entities in the transcription.

OCR4all (Reul et al., 2019) offers a text detection and OCR solution that is open-source and can be run locally. It offers two different approaches for text and layout detection: A connected component based approach using a configurable region template. In addition, recent versions of OCR4all[3] integrate a text line detection based on the kraken library (Kiessling, 2019). For the editing and correction step, OCR4all integrates LAREX as an editor. The layout editing workflow of LAREX is mainly focused around annotating regions of different layout elements on the page and text lines, which can be labelled using the region types and text types defined in the PAGE format. For the OCR correction, the user can either double-click on individual text line polygons to open a modal editor for this text line or switch to a text-line view, where text lines and their corresponding cutouts are alternatingly shown.

eScriptorium (Kiessling et al., 2019) provides an open-source solution to automatically recognize the layout and OCR of documents. It offers a layout editor based on baselines, lines and regions. The main layout correction workflow is built on the annotation of baselines, from which the resulting line polygons are automatically constructed. When making a correction to the transcription of a text line, a new window is opened, showing the line cutout and text field. In addition, a side-by side view reconstructing the original page's layout with the transcribed text or showing the text lines consecutively following the reading order is available. Tags can be applied to detected elements for semantic labeling. eScriptorium offers the ability to train new segmentation models to improve text detection for difficult document types, it doesn't contain an option to train the labels for individual lines or regions.

[1] https://www.loc.gov/standards/alto/ (last accessed: Oct. 23, 2025)

[2] https://ocr-d.de (last accessed: Oct. 23, 2025)

[3] https://github.com/OCR4all/OCR4all (last accessed: Oct. 23, 2025)

Callico (Kermorvant et al., 2024) is a flexible open-source framework for annotating documents and text in a collaborative workflow. The project focuses on a simplified user interface and is designed to generate data for machine learning efficiently. Individual steps in the annotation process are handled by individual tasks, which can optionally be split among users. The layout is annotated using labelled regions, but performed in a separate step from the text correction. Transcriptions are corrected by showing a split-view of the page and the text lines. In addition, the tool features the annotation of named entities and a "key-value" mode, which has been used to transcribe rows in handwritten tables. Although Callico doesn't integrate machine-learning algorithms directly in the application, it has been successfully applied for the annotation and correction of data for deep learning.

4. AdaptOCR: Annotation tool

For challenging transcription projects we introduce AdaptOCR, an efficient annotation tool for the presented workflow. In particular, we aim to improve upon existing editing solutions in the following general features:

- Efficient correction using a single pass for both layout and transcription
- Providing primarily a line-centric workflow with smart editing features
- Hierarchical annotation of regions with inheritable attributes
- Support for custom elements and attributes while being compliant with the PAGE standard
- Error-Detection for the transcription using a dictionary-based approach

The editor supports the annotation and correction of baselines, regions and text lines including their transcriptions. During the correction, two main views are utilized: A layout view for editing regions, text lines and their attributes and a text view, for correcting the transcriptions and reviewing the final annotations. Completed books can be exported either as PAGE, plain text, HTML or as a searchable PDF with the original images and a hidden text layer.

AdaptOCR is available as an open-source software[4] , that can be hosted on-premise or run locally using docker (Merkel, 2014). Because PAGE as the main representation is used, AdaptOCR can be easily integrated into different workflows and extended using other available open-source machine learning software. In the following, we present the features of AdaptOCR in detail.

4.1. Layout annotation

The first part of the transcription process of a page is the layout correction, as it determines the to-be-transcribed text elements. Therefore, all text lines must be annotated using bounding boxes or polygons in this step.

AdaptOCR shifts the focus to individual text lines instead of regions, as these are the cornerstone for the subsequent OCR. The page layout is visualized in AdaptOCR by overlaid rectangles or polygons for individual text lines on the page's image, which can either be added or corrected manually, similar to other editors such as LAREX, Transkribus, eScriptorium or Callico. By default, text lines are automatically mapped one-to-one to a PAGE "TextRegion" of their assigned type. Alternatively, multiple text lines might be grouped into a single TextRegion. Attributes, such as the language of the text can be set either on a line-level or on region level. Other region types are handled similarly to text lines: Their editable polygon is displayed and the region type and individual attributes can be set. For typing regions or individual line annotations, AdaptOCR offers a configurable set of classes which are highlighted in different colors to enable easy differentiation. Regions can be structured hierarchically to represent complex element arrangements, such as columns containing both text and image regions, or text regions containing handwritten artifacts.

Additionally, a data structure is available to store attributes on book-level, such as the main author of handwritten notes inside the book. These properties can also be inherited by pages and thus text elements, providing a default annotation, which can be overridden if necessary.

As some automatic processing pipelines recognizes text by detecting the baselines (including the integrated pipeline presented in section 4.3), manually annotating the baselines is also possible. If text is missed by the model, the user can correct the missing baseline and then re-run the text line segmentation directly from the editor interface to automatically infer the line's bounding polygon. This allows for an alternative baseline-focused workflow, which, might be beneficial to working with line polygons when the text detection is not sufficient and more baseline annotations are required for the fine-tuning of the text detection model.

To support efficient correction of errors, the editor offers several smart features. In line annotations built from baseline detections adjacent text lines are often horizontally concatenated. Therefore, dividing lines can be drawn to split multiple lines (e.g. when different columns were not correctly divided). To correct region polygons, e.g. for the training of

[4] https://github.com/uniwue-ai-ks/adaptocr (last accessed: March 14, 2026)

semantic segmentation models the editor offers a smart editing feature, which allows the user to draw a new polygon boundary. The editor than adjusts the appropriate polygons to fit the new boundary.

As discussed before, some documents might contain special elements like tables, mathematical or musical notations. As these elements require detailed annotation to be accurately represented, specialized annotation schemata are required. We argue that is not practical to implement a generic approach for annotating all the necessary different schemata. Therefore, to facilitate these workflows, such special page elements can be annotated as a corresponding layout element in AdaptOCR using a region of an appropriate type. The resulting PAGE should subsequently be loaded into specialized software for further and more detailed annotation.

4.2. OCR Correction

The second step in the annotation pipeline with AdaptOCR is the transcription of the text. The OCR editor is designed to reduce friction between the user and the application during the transcription as much as possible.

During the OCR correction, the page's image with highlighted text lines is shown on the left side. On the right side cutouts of individual text lines and their editable transcription is presented, as depicted in Fig. 1. During our transcription projects, we noticed that the addition of such an alternating view is superior to just a side-by-side view of the whole transcription and the page's image or individual line views, as the view stays more consistent and the transcription can directly be compared to the original. When a text line is selected or edited by the user, the corresponding area is highlighted in the original page's view on the left. This is especially useful for the annotation of handwritten artifacts, as they can be located anywhere on the original page. Yet, the transcription view is highly customizable, allowing the configuration of the segmentation view, the visibility of line cutouts etc.

To improve the recall of errors in high-quality OCR predictions, the editor integrates a real-time spell-checking function. The spellchecker is build on a base dictionary, for which any Hunspell[5] dictionary can be used, and an additional custom dictionary, which can be modified. During the transcription, words, that are not contained in the dictionary, are dynamically highlighted and suggestions for correct substitutions are shown. Incorrectly highlighted words can be removed from or added to the dictionary. Dictionary modifications are propagated in real-time to other users on the server, which simplifies collaborative work on the same corpus. In addition, a dictionary can automatically be learned from previously corrected transcriptions.

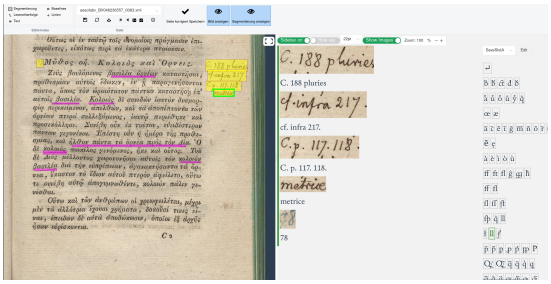


Figure 1: The transcription and proofreading interface of AdaptOCR. In the segmentation view on the left, handwritten text annotations are highlighted in yellow, handwritten line artifacts are highlighted in pink. Each recognized text line and its transcription are shown below each other in a list on the right.

When dealing with huge data sets, it's also important to offer a proofreading or reviewing functionality to the annotators to verify the work. Therefore, the editor supports showing the segmentation as well as the transcription in a single view. This enables a reviewer to efficiently go through the annotations and detect errors. Similarly to the layout correction view, different element types are highlighted using different colors and additionally important attributes can be displayed in the segmentation view or beside the transcriptions.

4.3. Trainable layout analysis and OCR

AdaptOCR offers integrated machine-learning models for both the layout recognition stage as well the OCR stage, which can be replaced by external models if appropriate. For the layout detection, we integrated the baseline detection model and text line segmentation algorithm by Fischer et al. (2023). Additionally, a rule-based layout recognition engine is integrated, which classifies detected text elements into different types: page header including page numbers, main text, footer (consisting of catch words or signature marks) and marginalia.

For challenging or unusual layout elements, custom layout detection models can be trained based on annotated pages using YOLO (Jocher et al., 2023). Then, during the layout recognition phase, these can optionally be used to segment and assign the learned types to the detected text or non-textual layout elements, such as handwritten lines. We used this approach in the detection of handwritten annotations, as presented below, to train a specific model to recognize handwritten text and line artifacts.

For the OCR, a fast text recognition model based on a CNN-BiLSTM + CTC architecture, is implemented using PyTorch (Paszke et al., 2019), which

[5] https://hunspell.github.io/ (last accessed: Oct. 23, 2025)

can be fine-tuned or trained from scratch by the user on either custom datasets or the corrected transcriptions from within the application.

An important feature of AdaptOCR is the ability to train and choose specific OCR models based on the predicted annotation type during the layout analysis. Therefore, a different OCR model could be used for handwritten artifacts than for the printed text on a single page. The differentiation can be made either by the type of the text region containing the text line, or by specifying required text line attribute values for each model. In a bilingual scenario (see below) this increased the accuracy of the text recognition and also the efficiency of the manual correction process with different experts for the two languages.

5. Comparison to other editing tools

An overview of the comparison of AdaptOCR with other tools is shown in Table 2.

For the layout annotation, AdaptOCR, similarly to Transkribus and eScriptorium, offers a complete feature set. In contrast to LAREX, it also supports custom attributes for different types. However, in Transkribus and eScriptorium custom attributes are represented as tags assigned to elements.

A special feature compared to the other editors is the invocation of automatic OCR after changes in the layout have been made. So, the consequences of the users layout corrections are directly reflected in the recognized text during the transcription step. This allows annotators to complete a page in one session without the need to exit the editor or run an OCR manually, which we believe greatly increases the overall productivity of the correction process. User feedback suggests that completing pages and therefore also books in a single pass feels more satisfying and productive than a workflow with multiple stages separated by potentially long time intervals.

For the text annotation, the user interface of AdaptOCR can be configured to either resemble the alternating cutout view of LAREX, or alternatively the side-by-side used in Transkribus, eScriptorium and Callico. However, AdaptOCR features no view reconstructing the original page's layout from the transcribed texts as available in eScriptorium. Among the presented editors, AdaptOCR is the only editor integrating a dictionary-based error detection for the transcription. However, other tools provide different means of error detection, such as eScriptorium or OCR4all's LAREX, in which text lines or words respectively are highlighted depending on the OCR engines confidence output in the transcriptions.

Regarding the automated layout detection, AdaptOCR offers an integrated pipeline for both the layout detection and the subsequent OCR. Here, the used models vary from platform to platform. Especially for the OCR, Transkribus offers substantial library of pre-trained models, which however, can only be used on the platform and cannot be downloaded and used locally. Each inference on the Transkribus platform requires payment in credits. In contrast to this, AdaptOCR, OCR4all and eScriptorium feature local training and models can be published and shared freely, which we believe is a key benefit to the community.

Because of the focus of AdaptOCR on the layout and text annotation, some features like named-entity annotation, which are present in eScriptorium, Callico and Transkribus are not available in AdaptOCR. Other features, like the Key-Value annotation in Callico, can however be achieved in AdaptOCR utilizing its flexible attribute annotation capabilities, although the user interface is not optimized for this use-case. In Transkribus, Key-Value annotation is available for metadata and named entities, but not for individual layout elements.

In order to achieve compatibility with other applications, both AdaptOCR and OCR4all store annotations directly in the PAGE format, whereas Transkribus can only export to PAGE. eScriptorium allows for importing and exporting annotations in the PAGE format. Because of this, it is possible to correct the layout in AdaptOCR and then utilize external PAGE-compatible OCR software, such as kraken or Calamari, for the text recognition step before correcting the transcriptions in AdaptOCR. For the latter, AdaptOCR has already been tested extensively. This also makes it possible to access already pre-trained models without the requirement of implementing custom interfaces for compatibility.

6. Application Scenarios

In this section, we present two projects using AdaptOCR and list their detailed requirements and challenges to the pages' annotation.

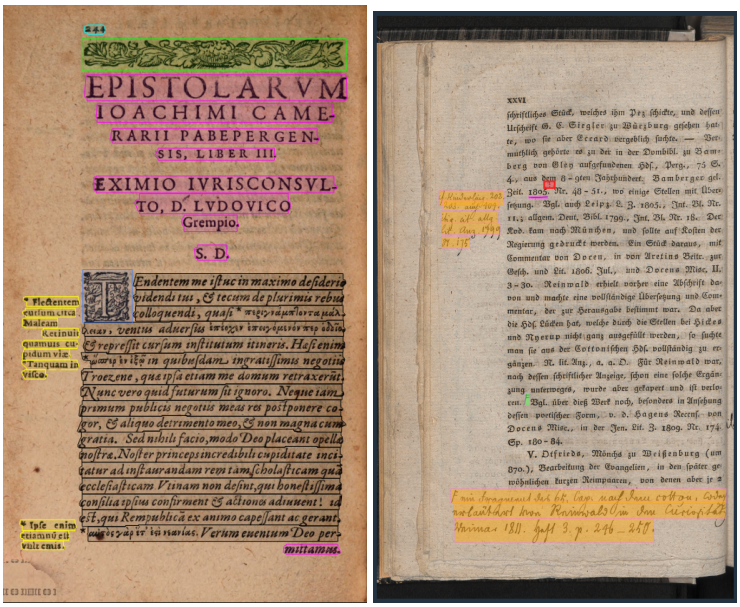
6.1. Transcription of historical printings

One real-world application of AdaptOCR was part of a larger project concerning the work of the medieval scientist Joachim Camerarius. As a result his books and letters (printed around 1500) with Latin and Greek text were transcribed, which comprised about 600 works with about 20000 pages in total.

For the layout analysis, the project used the pipeline presented by Fischer et al. (2023), which showed a recall of more than 99% for text lines and which is integrated into AdaptOCR. The first step of the pipeline is a pre-processing step, where the input pages are deskewed and scaled to a common resolution. Then, the baseline analysis is applied to detect the text elements on the page

Features / Application	OCR4all	Transkribus	Callico	eScriptorium	AdaptOCR
Open Source / Selfhosted	yes	no	yes	yes	yes
Layout Annotation					
Regions	yes	yes	yes	yes	yes
Text Line Polygons	yes	yes	yes	yes	yes
Automatic OCR on Layout Change	no	no	no	no	yes
Custom Region Types	no	yes	yes	yes	yes
Custom Attributes	no	tags	no	tags	yes
Text Annotation					
Dictionary Spell Checker	no	no	no	no	yes
Named Entity Annotation	no	yes	yes	yes	no
Key-Value Annotation	no	limited	yes	no	limited
Automatic Processing					
Baseline Detection	yes	yes	no	yes	yes
Textline Detection and Segmentation	yes	yes	no	yes	yes
Region Type Annotation	yes	yes	no	no	yes
OCR Training and Fine-Tuning	yes	yes	no	yes	yes
Training semantic layout labels	no	yes	no	no	yes

Table 2: Feature comparison matrix comparing (from left to right) OCR4all / LAREX, Transkribus, Callico, eScriptorium and AdaptOCR.



(a) Annotated page from the Camerarius corpus

(b) Annotated page from the Grimm corpus

Figure 2: Example from the editor's view for two pages from our projects. In (a) all text lines, drop capitals and image elements are labelled based on the text element's text type or region type. In (b) handwritten artifacts are annotated and highlighted in different colors based on the author and semantic meaning.

followed by a rule-based layout analysis recognizing headers, page numbers, marginalia, and footer regions. Furthermore, an instance segmentation model was trained for the detection of dropped capitals, which is similar to the layout training approach integrated in AdaptOCR. Fig. 2a shows an example of a page from the corpus annotated using our workflow. The resulting text lines are then transcribed using the integrated OCR engine of AdaptOCR. As Latin is the primary language with Greek being only present on less than half of the pages, often just as single words or phrases in the main text, two separate models were trained for Latin and Greek, which were automatically chosen during prediction, achieving a character error rate (CER) 0.4% and 0.76% respectively.

The workflow with automatic transcription, manual post-correction and use of the post-corrected texts to train a better model led to steadily improved recognition rates during the project as more training

data became available. In addition to the model, a Latin dictionary was also learned, which was primarily used to highlight all unknown words and errors in the hyphenation during the transcription, which significantly accelerated the post-correction process.

The final transcriptions are published in a searchable database[6] as a result of the project.

6.2. Annotation of Handwritten Artifacts

The overall goal of this project is to detect and transcribe all handwritten artifacts of the Grimm brothers in their private library (from around 1800 to 1850) containing more than 180000 pages. These handwritten artifacts are manifested as several lines of text, single words, letters or numbers, or line markings inside the printed text (e.g. underlines and correction marks). Since annotations are sparse with local accumulations, their detection is comparable to anomaly detection in business data.

The first step in the detection is to locate the baseline segments for any text lines and also bounding boxes for handwritten text and line artifacts. The artifacts are detected by a custom trained YOLO instance segmentation model. The second task is the transcription of the handwritten text, the assignment of the annotations to Jakob or Wilhelm Grimm or other persons, and optionally the assignment of the detected handwritten text markings to corresponding printed text.

In this project, AdaptOCR is being used to correct the handwritten annotations and lines, as well as their transcription. For this, the editor was configured to include different classes for each text element which distinguish between different sources of the handwritten annotation (annotation by Jakob or Wilhelm Grimm, other annotations). Furthermore, the detected line artifacts are annotated with different classes (underline, strike-through, markings, etc.). Because most annotations in one book are from the same person, the main author is set at book level and inherited by all handwritten text lines and only manually overwritten if necessary. Currently, approximately 20 different books containing ca. 3500 pages have been edited using AdaptOCR. From the collected data, we calculate the correction to take on average 9 seconds and 34 seconds for each drawn line or handwritten text artifact respectively, including the time required to add missing annotations, to adjust line polygons for training and to correct the transcriptions. Because of the difficult handwriting, several HTR engines and multimodal LLMs were applied and evaluated. Utilizing LLMs for the HTR shows a promising improvement: For longer handwritten notes on the cover pages of the books, where more context is available to the model, Gemini 3.0 Pro (Google, 2026) achieved a character error rate of 4.6%, resulting in a word error rate (WER) of 16.68%.

[6] https://texte.camerarius.de (last accessed: March 14, 2026)

7. Evaluation

In this section we present two evaluations quantifying the effectiveness the annotation workflow using AdaptOCR for the presented applications.

7.1. Dictionary Evaluation

We evaluated the effectiveness of the Latin dictionary support of AdaptOCR with two experiments in the context of transcribing Latin text in the Camerarius project. In the first experiment, we simply assessed the reduction of the word error rate (WER) of the prediction of the used OCR model, when only those words are corrected, which are not contained in the dictionary. The first line in Tab. 3 shows that for a corpus of 2000 pages (177942 words) with an initial WER of 3.53% of the OCR, the dictionary marked 4.51% of the words resulting in a WER of 0.82% after correction of these words. In the second experiment (second and third line in Tab. 3), we used two smaller corpora of 12 pages (3275 resp. 3296 words), which were transcribed with the same OCR model and manually corrected with and without the dictionary. With the dictionary, again only words not contained in the dictionary (6.86% of all words) were corrected which took ca. 16 minutes and resulted in a WER of 0.76%. Without the dictionary, all words were corrected, which took three times as long (ca. 48 minutes) and resulted even in a higher error rate of 1.07%. The disagreement between the annotator and the used ground-truth can be partly attributed to the task which required a diplomatic transcription including full replication of potential printing errors and partly to the historic printed source material, where words are often not separated clearly by spaces and ligatures as well as accents are difficult to distinguish.

7.2. Manual Correction Time

To evaluate the benefits of using the transcription view in AdaptOCR, we devised a small-scale pilot study. Two experiments were conducted on two sets of similar pages with printed Latin text in roman typeface from the Camerarius Corpus (the same as in the dictionary experiment) and on two sets of pages from the Grimm library containing dense handwritten texts from the Grimm brothers mostly written in German (see table 4). In both experiments, a small automatically transcribed corpus was corrected with the AdaptOCR proofreading view and a simple side-by-side view with a conven-

Dictionary	Corpus	# words	WER	Marked Words	WER after Correction	Time spent
yes	A	177942	3.53%	4.51%	0.82%	-
no	B	3275	2.32%	-	1.07%	48m 06s
yes	C	3296	2.12%	6.86%	0.76%	16m 25s

Table 3: Evaluation of the dictionary used in the Camerarius project. Corpus A is an automatically evaluated set of 2000 randomly selected pages, Corpus B and C are two manually corrected sets of 12 consecutive pages taken from two sample books from the Camerarius project. WER (word error rate) is measured in comparison to the final expert annotations produced during the project.

Annotation Tool	Corpus	pages	# words	WER	WER after correction	Time	seconds / edited word
AdaptOCR	Cam. C (pr)	12	3296	2.12%	0.70%	59m 23s	40.0
Side-by-Side	Cam. B (pr)	12	3275	2.32%	1.04%	1h 24m 45s	53.5
AdaptOCR	Grimm (hw)	11	1181	16.7%	-	53m 10s	16.2
Side-by-Side	Grimm (hw)	11	1280	13.9%	-	1h 0m 46s	20.5

Table 4: Results of a pilot study measuring the correction time in total and in seconds per edited word for the text correction using AdaptOCR's proofreading interface and a conventional text editor for documents from both the Camerarius (Cam.) and Grimm corpora. Cam B and C are the same pages as evaluated in Tab. 3. To provide initial transcriptions, for Camerarius, AdaptOCR's built-in OCR was applied to 12 pages each, for Grimm Gemini 3.0 Pro was applied to 11 pages each.

tional text editor displaying the page image adjacent to the transcription field.

In the first experiment, the two smaller corpora from table 3 were corrected without the dictionary by a different person. The first two lines in table 4 show, that the initial WER dropped from around 2.2% to only 0.7% resp. 1.04% due to the reasons stated above. However, the time necessary for the correction was about 25 % lower with the AdaptOCR proofreading view with a slightly lower WER after correction.

In the second experiment with the Grimm corpus (third and fourth line in table 4), the handwriting is difficult to interpret requiring an expert for the transcription. This is especially reflected in the high WER of the initial automatic transcription. For this domain, we also measured a reduction in the time required for each edit using AdaptOCR of about 21%. We believe this is a significant improvement, especially considering that the measured error rate was higher for the sample corrected in AdaptOCR and therefore more words needed correction in total compared to the sample corrected in a conventional text editor. Because reading the Grimm brothers' handwriting requires expert domain knowledge, the corrections could only be performed by one annotator. Therefore, no reference is available to compare the corrections against. Yet some words were still not readable and excluded from the evaluation.

8. Conclusion

We presented a workflow with AdaptOCR, a comprehensive editing solution for the transcription of challenging historical documents. The tool offers advanced features for the layout annotation and transcription correction, as well as the required flexibility to be applicable to a wide range of projects, as demonstrated in two challenging applications: the full transcription of printed text of a large corpus of historical printings and the detection / transcription of handwritten artifacts in the Grimm brothers' private library. We detailed how the underlying data format PAGE is used and provide comparison with existing tools highlighted AdaptOCR's features. In two experiments, we evaluated the possible efficiency gains using a dictionary-based workflow and compared the annotation speed in AdaptOCR's transcription view to a side-by-side view, showing a consistent improvement when using the transcription view.

We plan to use AdaptOCR for further transcription projects, including the transcription of historical handwritten parish documents, where the main challenges are the recognition of handwritten text and tables from different authors. Furthermore, we aim to extend the editor's integrated layout analysis algorithms and OCR models, not only to integrate newer and more advanced technology, but also to facilitate an easier integration of existing models using other text recognition engines including ensemble approaches. Additionally due to the recent advances of the OCR and HTR capabilities of large language models, we aim to integrate this new technology into the AdaptOCR workflow.

To conclude, we believe that our work represents a step towards improving the editorial workflow in transcription projects, supporting future research in the humanities.

9. Acknowledgements

We would like to thank Dr. Silvie Lang, University of Kassel, Germany, for her expertise and support regarding the transcription of the handwritten annotations by the Grimm brothers.

This work was in part funded by the German Research Foundation within the project *Referenzprojekt "Digitale Märchen-Handbibliothek von Jacob und Wilhelm Grimm"* (Deutsche Forschungsgemeinschaft(DFG) - Projektnummer 536456928)[7] and the project *Camerarius digital* (Deutsche Forschungsgemeinschaft (DFG) - Projektnummer 319239655)[8].

10. Bibliographical References

Denis Coquenet, Clément Chatelain, and Thierry Paquet. 2023. DAN: A segmentation-free document attention network for handwritten document recognition. *IEEE transactions on pattern analysis and machine intelligence*, 45(7):8227–8243.

Norbert Fischer, Alexander Hartelt, and Frank Puppe. 2023. Line-Level Layout Recognition of Historical Documents with Background Knowledge. *Algorithms*, 16(3):136.

Google. 2026. Gemini (version 3.0 pro) [large language model]. https://gemini.google.com. Accessed March 14, 2026.

Tobias Grüning, Gundram Leifert, Tobias Strauß, Johannes Michael, and Roger Labahn. 2019. A two-stage method for text line detection in historical documents. *International Journal on Document Analysis and Recognition (IJDAR)*, 22(3):285–302.

Mark Humphries, Lianne C. Leddy, Quinn Downton, Meredith Legace, John McConnell, Isabella Murray, and Elizabeth Spence. 2025. Unlocking the archives: Using large language models to transcribe handwritten historical documents. *Historical Methods: A Journal of Quantitative and Interdisciplinary History*, 58(3):175–193.

Adeela Islam, Tayaba Anjum, and Nazar Khan. 2023. Line extraction in handwritten documents via instance segmentation. *International Journal on Document Analysis and Recognition (IJDAR)*, 26(3):335–346.

[7] https://gepris.dfg.de/gepris/projekt/536456928 (last accessed: March 14, 2026)

[8] https://gepris.dfg.de/gepris/projekt/319239655 (last accessed: March 14, 2026)

Glenn Jocher, Jing Qiu, and Ayush Chaurasia. 2023. Ultralytics YOLO. https://github.com/ultralytics/ultralytics. Accessed October 23, 2025.

Philip Kahle, Sebastian Colutto, Günter Hackl, and Günter Mühlberger. 2017. Transkribus - A Service Platform for Transcription, Recognition and Retrieval of Historical Documents. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 04, pages 19–24.

Christopher Kermorvant, Eva Bardou, Manon Blanco, and Bastien Abadie. 2024. Callico: A Versatile Open-Source Document Image Annotation Platform. In *Document Analysis and Recognition - ICDAR 2024*, pages 338–353, Cham. Springer Nature Switzerland.

Benjamin Kiessling. 2019. Kraken - a Universal Text Recognizer for the Humanities. In *Digital Humanities 2019*, Utrecht, Netherlands.

Benjamin Kiessling, Robin Tissot, Peter Stokes, and Daniel Stökl Ben Ezra. 2019. eScriptorium: An Open Source Platform for Historical Document Analysis. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 2, pages 19–19.

Minghao Li, Tengchao Lv, Jingye Chen, Lei Cui, Yijuan Lu, Dinei Florencio, Cha Zhang, Zhoujun Li, and Furu Wei. 2023. TrOCR: Transformer-Based Optical Character Recognition with Pre-trained Models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(11):13094–13102.

D. Merkel. 2014. Docker: Lightweight Linux containers for consistent development and deployment. *Linux Journal*.

Tom Monnier and Mathieu Aubry. 2020. docExtractor: An off-the-shelf historical document element extraction. In *2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 91–96.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Stefan Pletschacher and Apostolos Antonacopoulos. 2010. The PAGE (Page Analysis and Ground-Truth Elements) Format Framework. In *2010 20th International Conference on Pattern Recognition*, pages 257–260.

Christian Reul, Dennis Christ, Alexander Hartelt, Nico Balbach, Maximilian Wehner, Uwe Springmann, Christoph Wick, Christine Grundig, Andreas Büttner, and Frank Puppe. 2019. OCR4all—An Open-Source Tool Providing a (Semi-)Automatic OCR Workflow for Historical Printings. *Applied Sciences*, 9(22):4853.

Christoph Wick, Christian Reul, and Frank Puppe. 2020. Calamari - A high-performance tensorflow-based deep learning package for optical character recognition. *DHQ: Digital Humanities Quarterly*, 14(2).

Yue Xu, Wenhao He, Fei Yin, and Cheng-Lin Liu. 2017. Page Segmentation for Historical Handwritten Documents Using Fully Convolutional Networks. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 01, pages 541–546.