READ

RECOGNITION & ENRICHMENT
OF ARCHIVAL DOCUMENTS

D4.6
Service and Tool Integration

Philip Kahle, Sebastian Colutto, Günter Hackl, Günter Mühlberger
UIBK

Distribution: http://read.transkribus.eu/

READ
H2020 Project 674943

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 674943

Project ref no.	H2020 674943
Project acronym	READ
Project full title	Recognition and Enrichment of Archival Documents
Instrument	H2020-EINFRA-2015-1
Thematic priority	EINFRA-9-2015 - e-Infrastructures for virtual research environments (VRE)
Start date/duration	01 January 2016 / 42 Months

Distribution	Public
Contract. date of delivery	30.06.2019
Actual date of delivery	10.07.2019
Date of last update	10.07.2019
Deliverable number	D4.6
Deliverable title	Service and Tool Integration
Type	Report
Status & version	FINAL
Contributing WP(s)	WP4
Responsible beneficiary	UIBK
Other contributors	All partners
Internal reviewers	Gundram Leifert, Hervé Dejean
Author(s)	Philip Kahle, Sebastian Colutto, Günter Hackl, Günter Mühlberger
EC project officer	Christopher Doin
Keywords	Transkribus

Contents

1 Executive Summary

This deliverable outlines the progress of task 4.2, service and tool integration.
During the first year of READ, procedures have been defined that allow standardized development and integration of tools from all technical partners.
In the second year, several of those applications could be tested and integrated in the platform while some improvements to the underlying architecture have been implemented.
Now, in the third year, new tools as the P2PaLA structure analysis were integrated into the platform, while others like Text2Image matching and HTR+ were re-integrated or updated. Those tools are listed and described in part two of this report.

2 Integrated Tools

2.1 Update of HTR integration: HTR+

The first tool that became available in the first year was the HTR engine provided by URO. This software is delivered in the form of one propietary jar file (Java archive) and an additional package by URO, the CITlabModule. The latter is made available via Github[1]. As this is Java software, it was easy to integrate within Transkribus, but, due to the complex nature of HTR technology, a lot of features have to be taken into account. The first experimental approach was a simple integration where only a provided HTR model and dictionary could be chosen by filename for recognition of a set of pages (where the layout must already exist). The training of new HTR models could be carried out manually by UIBK, i.e. no user interface was given for this approach. That was an important milestone, in order to gain knowledge about the said features of the technology but at the same time making the recognition available to experienced users.
In the beginning of year two, a user interface for training the HTR model was put in place which allows to configure and start this workflow: an arbitrary set of document images from a collection can be specified within Transkribus, while a nonintersecting set of images can optionally be specified as test set. Besides the parameters needed for the training, a user may specify a name for the HTR model, the language included and a description which are all stored as metadata in the database of Transkribus once the model has been trained. Furthermore, a list of known characters and the series of character error rate (CER) values, evaluated on the test and training set during the training process, are stored in the platform. All those properties shall outline the performance of the trained model very well and the training and test set can be viewed by any user, accessing the model, in order to gain an overview of the learned script type.
In year two, 383 HTR models (in the productive server, as of 28th of November 2017) have been trained by expert users via TranskribusX while the respective user interface and underlying engine have been improved over time. In the process, valuable experiences could be made by using this technology in production which allowed several

[1] https://github.com/Transkribus/CITlabModule

improvements to be made in the third year (see also section2.1.1).
A major update in early 2018 included the next iteration of the URO text recognition techology (see D7.9) which is called HTR+ in Transkribus. The underlying implementation now utilizes the Tensorflow framework[2] instead of the proprietary libraries and the training phase is massively shortened when a graphics processing unit (GPU) is provided. A training on the same dataset over 200 iterations within the Transkribus platform can now be carried out in less than half of the time while the recognition performance in terms of character error rate (CER) of the resulting model is drastically improved. This fact is outlined in table 1. Although HTR+ internally reduces the learning rate at the end of the training and thus a training with less iterations does not necessarily yield the same result, the CER values at the 100th iteration show that a good recognition result can be reached with fewer epochs in HTR+.

Technology	CITlab HTR	CITlab HTR+
Training time in hours	20:06	8:44
CER Train (epoch 100)	$16,13\%$	$3,73\%$
CER Test (epoch 100)	$15,13\%$	$2,90\%$
CER Train (epoch 200)	$10,51\%$	$2,53\%$
CER Test (epoch 200)	$10,08\%$	$2,63\%$

Table 1: Comparison of training processes for the models "Konzilsprotokolle M4", using the classic CITlab HTR, and "Konzilsprotokolle M4 HTR+". The training data consists of 54.306 lines and 350.067 words. Both trainings included 200 iterations. CER values after 100 and 200 iterations are given for test and train set.

Internal parameter tuning also made the user interface for configuring HTR trainings much simpler as only the specification of the number of epochs is required.
The HTR+ training feature was rolled out to Transkribus expert users in late 2018 and early 2019. To the date of this writing[3] 1.682 models have been trained with CITlab HTR and 962 with CITlab HTR+ by users in the Transkribus platform.

2.1.1 Management of Training Data

During the integration of and operation with three types of HTR technologies (UPVLC HMM HTR, CITlab HTR and HTR+), a lot of experience could be made with training data management for machine learning methods and it translated into improvements in the backend system as well as the user interface over time. While in the beginning the training of HTR models was a process operated by developers and adminstrators only, it is now available via the graphical user interface on request for interested users.
In earlier stages it became clear, that the state of the input data has to be captured precisely in order to get reproducible results but also to be able to track erroneous states

[2]https://www.tensorflow.org/
[3]30 June 2019

in the system and analyze them with the colleagues at the university of Rostock. The current training workflow therefore creates immutable and exact duplicates of the input transcription data while the original images, yet uneditable in Transkribus, are linked to it. The data then remains untainted by any change to the document data regarding segmentation, pagination, transcription text, etc., and eases reproducability of results as described before.
This method indeed fulfills the preservation requirements but also is subject to some disadvantages: the copy operations involved have a negative impact on space and time requirements. Training processes, especially those that include thousands of pages of training data, are excessively prolonged by this step. Also, the typical expert user's workflow in Transkribus comprises loops of training, recognition, correction and retraining with additional data where the preservation of the data is redundant to some extent and the reuse of training data is documented only implicitly via the image link.
An updated database model and training workflow that eliminate those shortcomings have been developed in the first half of 2019 and are currently being tested with a rollout planned for this summer. In the original data model of Transkribus transcriptions in the version history of document pages were bound to a page entity in a 1:1 relationship. This restriction was eliminated and the separate concept of a ground truth entity was introduced which allows to build relations between an image, a transcription (or even just a layout definition) and a trained entity, such as an HTR model. Merely building and storing those links eliminates the copy overhead and eases the reuse of training data. Adaptations in TranskribusX allow to browse and display existing ground truth data by model and select them for further trainings. As hinted earlier, this concept is not bound to HTR but can be reused for any training process involving user document data, such as the training of layout analysis technology.

2.2 Re-integration of Text2Image matching

The Text2Image tool to match a given text on page level that is not associated with any layout was already integrated into TranskribusX during 2017.
However many users experienced its usage as too complicated and overloaded for their specific task.
Also, the tool was not able to match a text based on an HTR+ model in the last iteration.
We have thus integrated a simplified version of the tool into TranskribusX that is able to choose HTR+ base models.
It can be accessed via the "Tools" tab in section "Other Tools". The user is now also able to start the matching process on a version of the transcripts with a specified edit status.
Also, it is possible to skip the layout analysis which enables a user to match text to a given layout that has for example been corrected by the user in beforehand.
Furthermore, several parameters for the matching were dropped or simplified to yield a cleaner interface.
We have also decided to drop the possibility to train a new model after a text matching for a certain number of iterations as from our experiments this functionality resulted in

little to no enhancement in the matching quality. It may be re-integrated into the tool in a later stage however if a successful use-case can be identified.

2.3 Integration of structure analysis tools: P2PaLA and dhSegment

Structure analysis is essentially the task of assigning a label (e.g. "paragraph", "heading", "foortnote") to a region in the layout of a document image.
During READ, two tools were developed that allow for such a structure analysis: P2PaLA from the UPVLC group and dhSegment from EPFL.
Both tools are available as OpenSource on GitHub:

- P2PaLA - https://github.com/lquirosd/P2PaLA
- dhSegment - https://github.com/dhlab-epfl/dhSegment

During the course of the last months, we have conducted experiments with both tools and decided to first integrate the P2PaLA tool on a recognition level into TranskribusX. It is accessible via the "Tools" tab in section "Other Tools".
The decision to integrate P2PaLA in favor of dhSegment was mostly due to the fact that the P2PaLA tool was able to read and write files in the exact format that is used by the Transkribus platform. Also, P2PaLA is streamlined to the exact usecase of structure analysis, while dhSegment is a more generic segmentation framework. However, an integration of dhSegment is also planned for future.
As said, we have integrated P2PaLA on the recognition level, which means that we are currently training models for the tool offline. The user can then select a model from a list and start the recognition process. We have also integrated a table that lists some useful information on the ground truth data of the currently integrated models.
In a future integration step, models may be associated to specific users and/or collections much like the way it is already implemented with HTR models.
Last but not least, the integration of the training step for P2PaLA may also be integrated in the way is is done for the HTR.

2.4 Cluster integration

The cluster implementation of the HTR+ recognition was less a concrete tool integration but an expansion and enhancement of the architectural capabilities of Transkribus.
We were able to integrate and parallelize the HTR+ recognition on the compute cluster LEO4 of the University of Innsbruck (cf. https://www.uibk.ac.at/zid/systeme/hpc-systeme/leo4/). It consists of 48 nodes with 28 Intel Xeon compute cores each, i.e. a total number of 1344 cores, where 44 nodes are equipped with 64GB of RAM.
The integration was done offline from Transkribus, that is data processed via LEO4 does not have to be uploaded to our Transkribus servers, but the images are just copied to a network drive and processed directly from there. This is due to the fact, that the cluster recognition is often performed for very large datasets, where an integration into

the main Transkribus system may not be feasible or necessary.
It is thus now possible to recognize a large number of files in a reasonable time due to the parallelism made available by the LEO4 cluster. This was already crucial several times when dealing with datasets that consist of thousands of pages as e.g. the case with the "Court Records" demonstrator by our partners at the National Archive of Finnland, where approximately 610.000 files were processed.

2.5 Error Rate Tool Update

For the evaluation of text recognition results Transkribus initially relied on the program *tasas*, which is included as standard tool in HTR software by UPVLC. In combination with a script for text extraction from PAGE XML, *tasas* allows to compare two fulltext versions: an hypothesis as given by the recognition and reference version, i.e. the ground truth. The computation returns a word error rate (WER) and a character error rate (CER). Although the computation is very quick, the tool sometimes produced results that did not match the validation results during CITlab HTR training and debugging the code or the underlying algorithm turned out to be too complicated.
As the mismatch was confusing to users, in 2018 tests were done with a library developed at URO, *CITlabErrorRate*, that allows to compare fulltext against a reference and returns several metrics, among them the well-known CER and WER[4]. Although the tool consumes slightly more memory at runtime, the results, comparable to the ones during validation, the extended number of metrics and the out-of-the-box multi-page comparison suggested to integrate this tool and take the place of the former solution.
In order to support very large input data sets, the computation now runs as an asynchronous process on the server-side and the data is stored persistently, which means that users can revisit former comparison results. The feature is accompanied with an extended user interface for the display of page-wise and overall results in a table as well as in a bar chart. The data can also be exported into Excel sheet for further analysis.

2.6 Sample Compare Tool Integration

Another potentially useful tool that is currently being integrated into TranskribusX is the sample compare tool, its purpose is to assess the quality of a model on an unseen dataset.
The tool is accessible in TranskribusX via the "Tools" tab under section "Compute Accuracy" and the button "Compare Samples".
Note that the tool is currently integrated on a proof-of-concept level, meaning that its usability can be greatly improved and some bugs have to be fixed.
The rough workflow of usage can be summarized as follows:
When clicking on the "Compare Samples" button the user first selects a set of documents for which the assessment should be applied to in the "Documents" tab of the appearing dialog. In this dialog the user also sets the number of lines that are randomly extracted from those documents (the default is 100).
Clicking on "Create Sample" starts a job that extracts those lines and stores them as a

[4]https://github.com/CITlabRostock/CITlabErrorRate

new document, one page for each line, into the selected collection.
This sample document, i.e. the randomly selected lines, should then be corrected by the user to generate a ground-truth for the comparison.
Afterwards, the user has to perform an HTR recognition process on the lines with the model that shall be assessed for the document set where those random lines were taken from.
Finally, the user can re-enter the "Compare Samples" dialog, select the generated sample under the "Samples" tab and compute the predicted accuracy of the model.

3 Conclusion and Outlook

As outlined, in 2018 and 2019 many important tools were integrated and updated according to our workplan from last year.
However, as the project came into its final phase, we also realized that we have to streamline the integration of tools to the specific task that is ahead of us: the successful foundation and implementation of a European cooperative.
We have thus decided to drop several tool integrations that are in our opinion not crucial to the success of the cooperative, partly because of their unstable academic nature, partly because they do not fit the main objective of the coop, that is Handwritten Text Recognition and Enrichment.
As an outlook to our next steps, we are planning to further integrate and fine-tune the structural analysis tools, as we think (and as the large interest of several experienced transcribers already proves) that this kind of enrichment to the documents is of very high interest to many users.
Last but certainly not least the integration of the second HTR engine produced during the READ project, namely (Py)Laia (https://github.com/jpuigcerver/PyLaia) from the UPVLC group is an important task ahead that should positively stimulate the competition between HTR providers in the Transkribus cooperative.