ocr4all

OCR for Incunables

by Johann Ramminger

Probably the most significant step forward for quantitative (really any kind of text-oriented) research in Early Modern Latin (EML) in a long time is **ocr4all**, an OCR software that reliably converts scans of early printed books to machine-readable (and human-researchable) text, developed at the U. of Würzburg (github.com/OCR4all). High quality scans of early printed books have been abundant for some time now; that has, however, so far not translated into an increased availability of texts.

ocr4all

ocr4all is designed for the single user without extensive IT-knowledge or -support (i.e. me). I have only used it with Latin texts, but the official test case is a multilingual textcorpus from the 15th/16th century, Narragonien digital; ocr4all works independently of language and script.

Please note that the following comments refer to the version of May 2019; as of fall 2019 there will be a new version with substantial (?) changes, which may behave differently (I have so far only read the new German help file with the version number 2.1).

Setup

ocr4all comes with a setup-guide that simply works - which is not my experience with open source software generally. Details: I have only ever installed the Docker Toolbox. Firstly, during the installation of the Toolbox, you are presented with optional installations: Despite this being seemingly an option, you will need Git for Windows. Kitematic I have not installed. On one of four machines I had to update the Oracle Virtual Box during install. This is straightforward. Secondly, the following seems an option in the *ocr4all* setup guide ("if you want to"), but setting the (green) maximum on both memory and cpu-cores of the Virtual Machine, as described in the guide, is an absolute must. I started out with Windows 7 and switched to Windows 10; as far as I can tell there is no difference in use.

Use and Results

ocr4all has a graphical user interface and runs in the browser; I use *Chrome* as recommended.

Quality of the Scans

Where to find scanned books: Google and archive.org are of course your friends, but for incunables you should consult ISTC and GW, both of which have references to scans (and here you can be sure that they have not been misidentified). *ocr4all* is not demanding, but of course the cleaner a scan the better. There can be unexpected obstacles. I was looking for a 1470-Plutarch, and found a beautiful scan in the HAB in Wolfenbüttel. The only problem: the proud first owner had marked every capital letter with a red stroke. Via the ISTC I found a high-quality scan of a pristine copy in the Marciana. Even though I have often read Sweynheym & Pannartz prints from the early 1470s on paper, I never realized that their font lacks i-dots; that did not help (at least that was my experience; probably more training would have helped). As soon as you get into the 1480s, the typeface becomes rapidly less of a factor, and a print by Froben or Jean Petit will be easily legible for *ocr4all*. I have not yet tried any of Aldus' books set in the 'Aldine' italics.

Preparation of the pages

ocr4all copes with sophisticated types of layout, columns, inserted graphics, etc. I have tried to eliminate problems in advance by preparing the pages with *ScanTailor* (deskew, split columns, remove marginalia) and *IrfanView* (cut out graphics, initials). To avoid degradation through multiple conversions I work with tiffs as much as possible, even though they need space, and convert to png only before *ocr4all* (*IrfanView* is a whiz at converting formats). Alternatively, to save the last step when I am feeling lazy, I just put any scan into the 'input'-folder, and ocr4all will convert it when loading the project (but will reset the pages numbers from 1).

Training

ocr4all has a pretrained set of generic font models which you can download from the github-page under ocr4all-models (it is able to read fraktur, see below). Recognition accuracy begins in the 96% percentile, but will get up to 99% percent, if you invest time in training the model on your specific typeface. Three to four pages, twice repeated with different pages, were enough to bring the 1500 *Apuleius commentary* of Beroaldo to over 98% accuracy, which means about an hour's investment. The time you invest in repeated training to bring the accuracy up to over 99% you will save during the post-processing correction. It took me some time to arrive at that insight, but even in that percentile enough mistakes remain - and if you are working with incunables, they often are faulty or badly legible to start with. I have made no attempt at training or recognizing Greek. It may be worth your while to select the pages for training with care if you have a text where the letters ‘y’ (sometimes recognized as ‘v’) and ‘z’ (often not recognized at all) are important. I found this out the hard way with the *Antiquitates* of Annius of Viterbo (Romae 1498), where toponyms with ‘z’ got mangled. The problem is to find the right pages, since few words have a ‘z’.

Fraktur

Update 19 September 2019: I am in the process of ocr’ing a book in Gothic type (Annius da Viterbo, De futuris Christianorum triumphis, Genova 1480, a perfect scan from archive.org), and the results exceed my wildest expectations, even before training. Weaknesses before training: ‘et’ (which resembles ‘7’) recognized rarely, some capital letters not recognized, final m (‘3’) read as ‘z’, ‘b’ and ‘h’ often confused (unfortunately they look the same, so in this case I have little hope for training).

Use

There are lots of possibilities to tweak the program, but so far I have been content with the default settings. The only problem that returns consistently concerns segmentation. In cases where two short lines follow each other (e.g. end of paragraph and following headline) their sequence can be inverted. Also lines which contain longer whitespace may be split and the parts reordered (this can make the ocr of the indices of incunabula into a pitiful mess). In all these cases the official solution is to recombine the lines and put them in their original order one by one using LAREX. This involves a significant expenditure of time. I have so far done without the indices, and recombined and reordered the lines of normal text (as far as I have noticed the mistake) by hand in postprocessing.

ocr4all does not use a dictionary (as OCR programs for modern texts usually do) - which for EML does not exist - , but uses a ‘short memory’ approach which I cannot claim to understand, but which works. On the average office computer *ocr4all* is not fast, fifty pages of a folio-volume will certainly take me a couple of hours, but the program runs happily in the background. My private laptop as well as my office PC are senior citizens, so yours may be faster.

Result

The result is a text file (unicode) which contains a precise representation of the original, line for line. *ocr4all* distinguishes reliably (and without training!) between ‘f’ and the ‘long s’ (which makes it a prime choice for books up to the 1850s, the typeface of which may otherwise be unproblematic). Abbreviations are rendered as on the printed page, *ocr4all* makes no attempt at guessing what an abbreviation might stand for. Anybody who has ever been puzzled by ‘aim’, ‘aunt’ (for ‘animi’, ‘autem’) and similar monsters will be grateful for this.

Postprocessing

Since I have mostly dealt with books with extensive and ambiguous (per/par, con/com!) abbreviations and lots of words split without hyphens, postprocessing (search-and-replace) has normally taken me more time than ocr’ing the texts. I am developing scripts for solving abbreviations and spell-checking Early Modern Latin tailored to *ocr4all*, which eventually will be available to fellow users. Actually, I look forward to a few seventeenth century or later books without abbreviations.

Ease of Use

I did not find the learning curve steep (except for the LAREX-tool for layout analysis which I have not mastered). Initially there were glitches (mostly probably due to the limitations of my elderly PC) which in some cases could only be solved by a restart; but I have not had a problem since the May 2019 update. Even in the earlier version, I never actually lost any work, *ocr4all* seems to write everything to the harddisk immediately.

At the moment, I have the same sense of wonder I had when I first entered the Vatican Library many years ago: when it slowly dawned on me that just about everything I might want to read was within arm’s length. My research nowadays largely depends on machine-readable EML texts, and the major limitation, that many texts simply are not available, has now been removed.

Literature

Christian Reul, Uwe Springmann, Frank Puppe, LAREX – A semi-automatic open-source Tool for Layout Analysis and Region Extraction on Early Printed Books. In Proceedings of the 2nd International Conference on Digital Access to Textual Cultural Heritage (2017). arxiv.org/abs/1701.07396 with further literature.

Uwe Springmann, Christian Reul, Stefanie Dipper, Johannes, Baiter, GT4HistOCR: Ground Truth for training OCR engines on historical documents in German Fraktur and Early Modern Latin. Preprint on www.researchgate.net, dataset GT4HistOCR on zenodo.org (dataset under CC-BY 4.0 license).

I will try to keep this post up to date with my experiences. Last updated: 20.09.2019

Written on May 10, 2019