READ
RECOGNITION & ENRICHMENT
OF ARCHIVAL DOCUMENTS

D7.21
Model for Semi- and Unsupervised HTR Training P3

How to get a good HTR without manual ground truth production

Gundram Leifert, Roger Labahn
URO

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	31.12.2018
Actual date of delivery	30.12.2018
Date of last update	December 19, 2018
Deliverable number	D7.21
Deliverable title	Model for Semi- and Unsupervised HTR Training P3
Type	Demonstrator
Status & version	final
Contributing WP(s)	WP7
Responsible beneficiary	URO
Other contributors	UPVLC,UCL
Internal reviewers	Joan Andreu Sánchez (UPVLC), Tobias Hodel (StAZH)
Author(s)	Gundram Leifert, Roger Labahn
EC project officer	Christophe Doin
Keywords	semi-supervised, text alignment, text2image, HTR, training

Contents

Executive summary

The third year deliverable describes the task and generic sub-tasks of it. After a first proof of concept in year two, in this year we can show that the algorithms work reasonable well for very challenging real-world problems. We show how to handle a real-world scenario with the example of letters from the British philosopher Jeremy Bentham, provided by our partner University College London (UCL).

1 Introduction

As mentioned in Del 7.19 and 7.20, the general strategy to use already existing transcripts to automatically create so-called *ground truth* (GT) to train a Handwritten Text Recognition (HTR) model is shown [2][3].

In real-world scenarios some new problems appear, that have to be solved by the text2image process (see [3] for details). The transcriptions provided to the semi-supervised workflow are problematic (in some cases false, in some cases according to specific transcription guidelines not directly usable) and the recognition of the layout of text lines can also be unreliable.

In the following we want to show how to handle the large amount of unassigned transcripts with their corresponding images and how they can be used to train an HTR. In Section 2 we will describe the data. In Section 3 we will prepare the input so that it can be used in Section 4 for the text2image process. Finally we will apply a modified semi-supervised training workflow in Section 5 using the components introduced in Section 3 and 4.

2 Data of Jeremy Bentham

Jeremy Bentham was born in London in 1748 and died in 1832. He devised the doctrine of utilitarianism, arguing that the ‘greatest happiness of the greatest number is the only right and proper end of government’. He was a major thinker in the fields of legal philosophy and representative democracy, and originated modern ideas of surveillance through his scheme for a Panopticon prison. He supported the idea of equal opportunity in education and his ideas contributed to the foundation of University College London in 1826, the first institution in England to admit students of any race, class or religion and the first to welcome women on equal terms with men.

In the period from 1760 to 1832 Jeremy Bentham and his assistants wrote approximately 75,000 folios: 60,000 of these are held in Special Collections at University College London and 15,000 are in The British Library. Around 31,000 pages of this material has been transcribed using the well-known Text Encoding Initiative (TEI) XML mark-up to tag features of the manuscripts. Volunteers in the Transcribe Bentham crowdsourcing initiative[1] have transcribed around 21,000 of these pages; with the remaining 10,000

[1] Transcribe Bentham: http://transcribe-bentham.ucl.ac.uk/td/Transcribe_Bentham

pages transcribed by researchers at the Bentham Project[2], the world centre for Bentham studies.

3 Text2image Preparation

To run the text2image process the transcripts and images have to be prepared. In this section we will convert the transcripts XML with markups into raw text strings and the images will be processed by a Layout Analysis (LA) and HTR.

3.1 Line-ConfMat Calculation

The text2image process needs the so-called Line-ConfMats from text lines of the image [3]. These Line-ConfMats contain a confidence that a specific character occurs at a dedicated position in the text line. To calculate them, as first step text lines have to be detected by an LA (see green baselines in Figure 1a).

The resulting order of the text lines does not necessarily fit to the order of the transcribed lines. In our example the LA interpreted the three text lines on the left side in Figure 1a as marginalia and put them into another text region. As result, these three text lines are ordered in front of the text lines in the main body, whereas their corresponding transcripts occur in between the main body transcripts.

As last step the HTR reads the text lines given by the LA and calculates the Line-ConfMats.

3.2 Transcript Preparation

The goal is to extract possible text line transcripts for the text2image process. In general the transcriptions are done for another project or goal so that we always have to clean up the transcripts so that they can be used for the text2image process. In our case the transcripts are marked up with TEI-XML tags (see Figure 1b) and have to be converted into *raw text lines* (see Figure 1c). The following markups have to be handled:[4]

del The text is stroked through, but still was readable for the transcriber. The transcription is correct, so we can use transcripts with these tags as raw text line.

add The text is added (mostly above) a normal text line. An LA either does not find theses addition or finds those as an additional line. We use the transcripts of these tags as additional separate raw text line.

sic The transcriber found a typo and marked it. We want the HTR to transcribe the characters how they are written, so we leave the transcripts unchanged in the raw text line.

[2] The Bentham Project: https://www.ucl.ac.uk/bentham-project

[3] Source code publicly available at https://github.com/CITlabRostock/CITlabConfMat

[4] See http://transcribe-bentham.ucl.ac.uk/td/Help:Transcription_Guidelines for more information

note Notes are mainly marginalia left or right of the text main body. We interpret them as separate raw text lines.

unclear If the transcriber is unsure about the spelling, the probability of a typo by the transcriber is too high. Lines containing these markups are ignored.

foreign Whereas most of the transcripts are English, non-English words are marked with this tag. The HTR should be able to transcribe these characters as well. Thus, the transcripts in this tag are used.

hi If the author underlines characters or puts characters in superscript, the HTR should also transcribe these characters, so the transcripts in these tags are used for the raw text line.

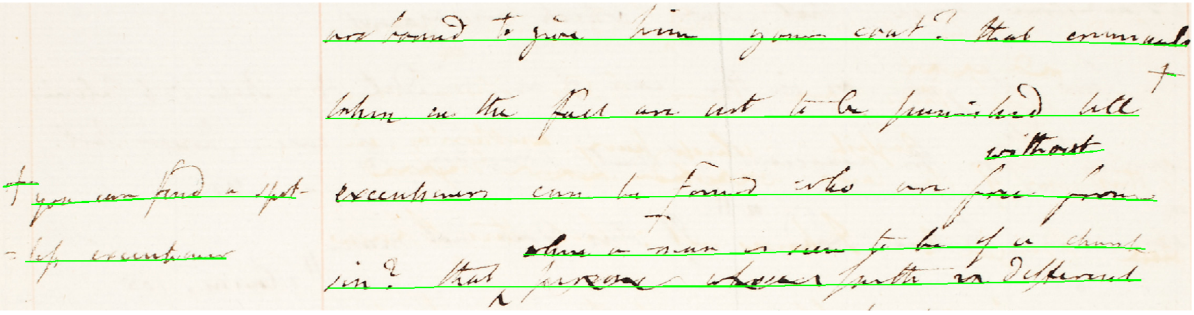
gap If the transcriber is not able to read the stroked-through text, he adds this tag. For the HTR there is missing a correct transcription, so we skip lines with these tags.

As additional transcription rule, UCL instructed the transcriber to transcribe hyphenated words as whole word (see Figure 1a left side and Figure 1b in the *<note>*-tag). So a line break denoted by "*<lb/>*" do not necessarily fit to the line breaks in the image. In addition, each word may be hyphenated. This has to be taken into account in the text2image process.

4 Text2image Process

The list of Line-ConfMats and the list of cleaned transcripts are the input of the text2image process. As already described in [2], we concatenate all Line-ConfMats by adding a row in between which has the probability of 1 for the synthetic character "↵". In the same way we concatenate the line transcripts using the line feed character "\n". Now, we have one long ConfMat containing all text lines and one long transcription, both related to the same page. We can use an approximation of CTC-algorithm (see [1] for details) to calculate the probability of the transcript in this ConfMat. If one only uses the most probable path (without summing up paths), one gets an exact assignment between the ConfMat and the transcripts. This path we call *BestPath*.
For pages with large amount of text lines the calculation of the mapping gets so large that a naive approach for solving the assignment is not possible. Therefore we transform the calculation of the BestPath into a so-called "shortest path problem" and use the Dijkstra algorithm to efficiently find the shortest path in the resulting directed graph [4][5]. This graph can be embedded into the 2D space, whereas the y-dimension is us used for the position in the ConfMat and the x-dimension for the position in the transcripts. For each edge of the Graph in position (y, x) we can calculate the costs to assign the first x characters to the ConfMat being in ConfMat position y. The Graph can be visualized using the costs of each edge as heat map. For our example text alignment task with text lines from Figure 1a and transcripts from Figure 1c the resulting heat map is shown in

[5] Source code publicly available at https://github.com/CITlabRostock/CITlabTextAlignment



are bound to give him your coat? that criminals
taken in the fact are not to be punished till +
without
executioners can be found who are free from
where a man is seen to be of a church
sin? that ~~persons whosoever~~ faith ~~in~~ different

+ you can find a spot-
-less executioner

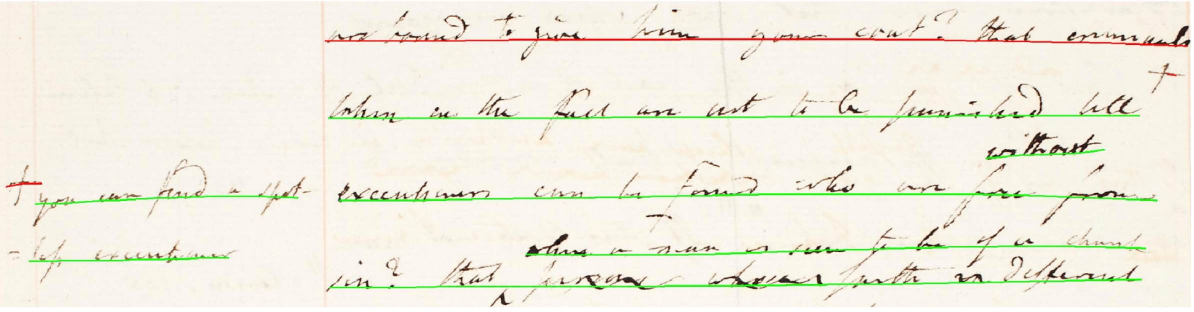
(a) **LA result.** The pages contain many notes, additions and stroke throughs, which have to be found by the LA (green lines).

```
                    <lb/> are bound to give him your coat? that
criminals<lb/> taken in the fact are not to be punished till <
hi rend="superscript">+</hi><note><hi rend="superscript">+</hi>
 you can find a spotless executioner</note>executioners can be
found who are free <add>without</add> from <lb/>sin? that <del>
persons whosoever</del><add>where a man is seen to be of a
church</add> faith <del>in</del> different<lb/>
```

(b) **XML transcripts.** The transcripts corresponding to Figure 1a contain many markups, but also missing hyphenations (see ‘spotless‘, which is "spot-" and "-less" in the image).

```
          are bound to give him your coat? that criminals
          + you can find a spotless
          executioner
          without
          taken in the fact are not to be punished till +
          executioners can be found who are free from
          where a man is seen to be of a church
          sin? that persons whosoever faith in different
```

(c) **Raw text lines.** The transcripts of Figure 1b are prepared for the text2image process according to the rules mentioned in Section 3.2.



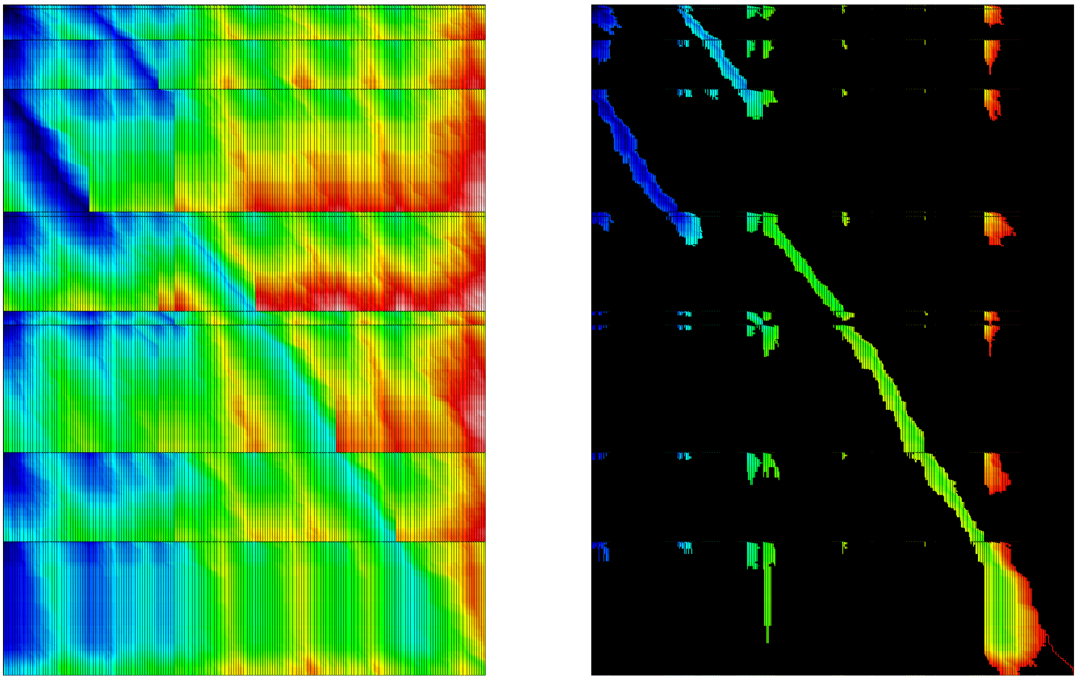
are bound to give him your coat? that criminals
taken in the fact are not to be punished till +
without
executioners can be found who are free from
where a man is seen to be of a church
sin? that ~~persons whosoever~~ faith ~~in~~ different

+ you can find a spot-
-less executioner

(d) **Result of text2image.** Only for the first line and the both "+" signs the algorithm cannot find corresponding transcripts so that they cannot be used as training samples (red lines).

Figure 1: **Example text2image process on part of image 100_019_002**: From the XML transcripts the raw text lines were extracted. The LA detects the text baselines, which are candidates for training samples. The text2image process tries to assign the transcripts to the raw text lines.

Figure 2a. Transforming the alignment task in a shortest path problem, we can take advantage of the Dijkstra algorithm and graph properties so that we do not have to follow paths that cannot result in the shortest path. Depending on LA, HTR and transcription quality as well as the graph properties, the number of vertices to calculate can be reduce to less than 5 %. In Figure 2 the reduction of calculation is shown.



(a) **Naive implementation.** Each edge have to be caluclated.

(b) **Efficient implementation.** ~7.5 % of the edges have to be calculated (colored)

Figure 2: **Dynamic programming to solve the text2image process for the example of Figure 1.** Using the Dijkstra algorithm and other properties of the graph, the number of vertices, that have to be calculated to get the shortest path, is reduced from 1,686,508 to 124,871.

5 Semi-Supervised Training Workflow

In this section we will apply the semi-supervised training workflow as described in [3, Section 2.1] with the modification tha no manually transcribed and aligned (to the image) GT is available (compare Figure 3 and [3, Figure 2]). For this reason a *base HTR* model has to be provided for the initial text2image process. The rest of the workflow remains unchanged whereas the HTR is trained without manually produced GT. As base HTR we choose a model trained on very different scripts, written by Queen Charlotte (1744 – 1818). As expected the quality of the HTR is very bad with more than 40 % Character Error Rate (CER) (see the initial CER in Figure 4).
We will test the quality of the HTR on two test sets provided by UCL. Both sets

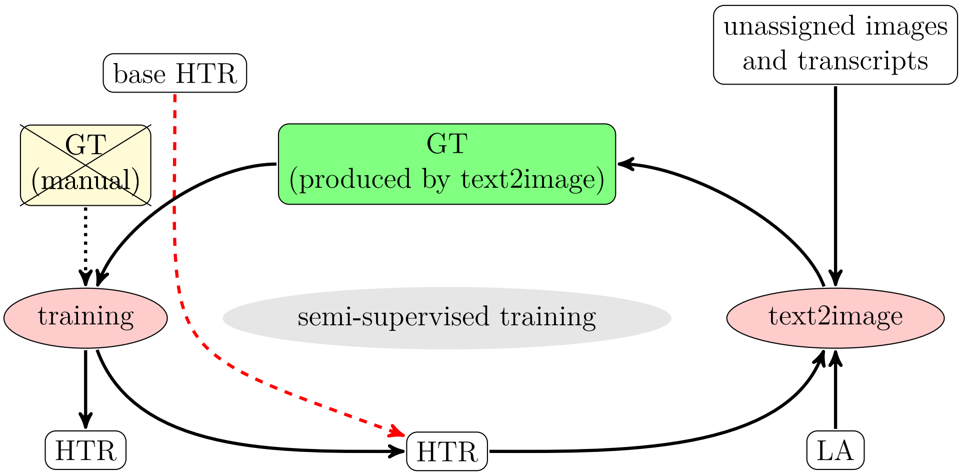
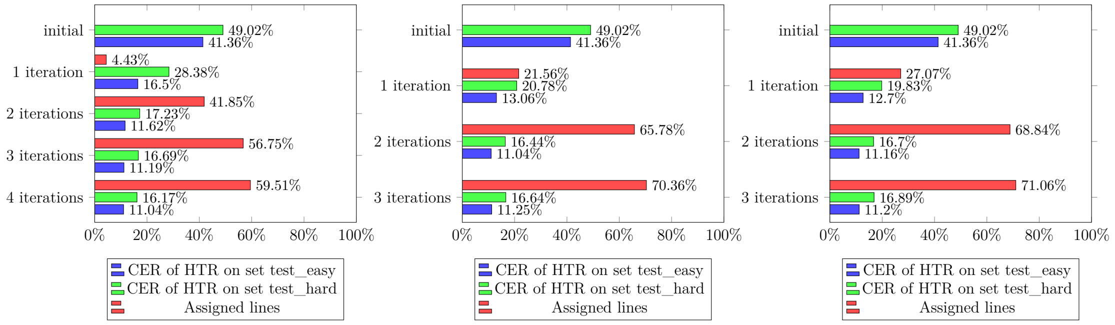


Figure 3: **Semi-supervised training workflow without manually produced GT:** Instead of training an HTR on manually produced GT, for the initial text2image process a *base HTR* is taken. For the next iterations the HTR resulting from the training is used. After each iteration the HTR becomes better and the text2image process can provide more GT.

are part of the Bentham collection and are available in Transkribus in Collection-ID 27931 as Document-ID 105016 (test_easy) and 105019 (test_hard). We will apply three iterations of the semi-supervised training workflow to train an HTR. Therefore, we will leave all hyperparameters fixed except for the *confidence threshold* $t = 0.1$, $t = 0.01$ and $t = 0.005$, specifying when to accept a text line with its assigned transcript as training sample. There is a trade off between quantity and quality of the training samples: The higher the value t, the more confident the matching will be, but the fewer training samples will be provided.

For the text2image process, the LA found 231,855 lines on 3,869 pages. These lines all are candidates for training samples, whereas many of them are wrong or not found



(a) *Threshold = 0.1* with 4 iterations

(b) *Threshold = 0.01* with 3 iterations

(c) *Threshold = 0.005* with 3 iterations

Figure 4: **Quality of text2image and HTR in the semi-supervised training workflow**. The amount of training samples depends on the HTR quality and the threshold t

properly. Depending on the threshold t we obtain GT to run the training process (see Figure 4). The training process leads to an HTR which again can be provided for the text2image process to generate better/more GT. The base HTR has a CER of 41.36% on the test_easy set. The text2image process, suffering from the bad adapted HTR model, is able to find only 10273 (= 4.43%) lines for $t = 0.1$ (see Figure 4a). Nevertheless, the HTR trained on these lines comes down to a CER of 16.50% on the test_easy set. In the second iteration the test2image process can find 97022 (= 41.85%) lines due to the better HTR. The subsequent training process leads to an HTR with a CER of 11.62% on the test_easy set.

Comparing Figures 4a and 4c, the semi-supervised training process is very robust against the threshold, but has an effect on the converging speed. So the training process can better deal with faulty transcriptions than with too few training samples.

With successive number of iterations the obtained training samples and the quality of the HTR increases, whereas a saturation is observed within two to 4 iterations with CER $\approx$ 11% for all thresholds. If enough GT is available it is better to train the HTR only on the more confident training samples to not obtaining too many alignment errors in the training samples (compare 4a and 4c in the last iteration). If we apply the classical training on 21,744 clean manually transcribed and aligned GT lines, the resulting HTR reaches a CER of 17.39% on test_hard set and 9.1% on the test_easy set. Thus, on the test_hard set the semi-supervised training performance already is competitive to the classical approach. In future our aim is to improve the semi-supervised training so that we can outperform the classical training with aligned GT if we have ~10 more unaligned GT.

In conclusion, we have shown that an HTR can be trained with the semi-supervised training workflow even if the GT is not aligned to the lines. Thanks to accepting only training sample that were matched with a sufficient confidence, even challenging layouts and erroneous transcripts can be used in the semi-supervised training workflow.

References

[1] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural nets. In *ICML '06: Proceedings of the International Conference on Machine Learning*, 2006.

[2] T. Grüning, G. Leifert, T. Strauß, and R. Labahn. D7.19 model for semi- and unsupervised htr training p1. approaches, scenarios and first results. Technical report, READ EU project (674943), 2016.

[3] G. Leifert, T. Strauß, and R. Labahn. D7.20 model for semi- and unsupervised htr training p2. how to get a good htr without expensive ground truth production. Technical report, READ EU project (674943), 2017.

[4] Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Science & Business Media, 2003.