Originally published in: Demner-Fushman, Dina/Cohen, Kevin Bretonnel/Ananiadou, Sophia/Tsujii, Junichi (Eds.):
Proceedings of the 20th Workshop on Biomedical Language Processing. June 11, 2021.
– Stroudsburg: Association for Computational Linguistics, 2021. Pp. 168-179.
DOI: *https://doi.org/10.18653/v1/2021.bionlp-1.19*

Word-Level Alignment of Paper Documents with their Electronic Full-Text Counterparts

Mark-Christoph Müller, Sucheta Ghosh, Ulrike Wittig, and Maja Rey
Heidelberg Institute for Theoretical Studies gGmbH, Heidelberg, Germany
{mark-christoph.mueller,sucheta.ghosh,
ulrike.wittig,maja.rey}@h-its.org

Abstract

We describe a simple procedure for the automatic creation of word-level alignments between printed documents and their respective full-text versions. The procedure is unsupervised, uses standard, off-the-shelf components only, and reaches an F-score of 85.01 in the basic setup and up to 86.63 when using pre- and post-processing. Potential areas of application are manual database curation (incl. document *triage*) and biomedical expression OCR.

1 Introduction

Even though most research literature in the life sciences is *born-digital* nowadays, manual data curation (International Society for Biocuration, 2018) from these documents still often involves paper. For curation steps that require close reading and markup of relevant sections, curators frequently rely on paper printouts and highlighter pens (Venkatesan et al., 2019). Figure 1a shows a page of a typical document used for manual curation. The potential reasons for this can be as varied as merely sticking to a habit, ergonomic issues related to reading from and interacting with a device, and functional limitations of that device (Buchanan and Loizides, 2007; Köpper et al., 2016; Clinton, 2019).

Whatever the *reason*, the *consequence* is a two-fold media break in many manual curation workflows: first from electronic format (either PDF or full-text XML) to paper, and then back from paper to the electronic format of the curation database. Given the above arguments in favor of paper-based curation, removing the first media break from the curation workflow does not seem feasible. Instead, we propose to bridge the gap between paper and electronic media by automatically creating an alignment between the words on the printed document pages and their counterparts in an electronic full-text version of the same document.

Our approach works as follows: We automatically create machine-readable versions of printed paper documents (which might or might not contain markup) by scanning them, applying optical character recognition (OCR), and converting the resulting semi-structured OCR output text into a flexible XML format for further processing. For this, we use the multilevel XML format of the annotation tool MMAX2[1] (Müller and Strube, 2006). We retrieve electronic full-text counterparts of the scanned paper documents from PubMedCentral® in `.nxml` format[2], and also convert them into MMAX2 format. By using a shared XML format for the two heterogeneous text sources, we can capture their content and structural information in a way that provides a *compatible*, though often not identical, word-level tokenization. Finally, using a sequence alignment algorithm from bioinformatics and some pre- and post-processing, we create a word-level alignment of both documents.

Aligning words from OCR and full-text documents is challenging for several reasons. The OCR output contains various types of **recognition errors**, many of which involve special symbols, Greek letters like μ or sub- and superscript characters and numbers, which are particularly frequent in chemical names, formulae, and measurement units, and which are notoriously difficult for OCR (Ohyama et al., 2019).

If the printed paper document is based on PDF, it usually has an **explicit page layout**, which is different from the way the corresponding full-text XML document is displayed in a web browser. Differences include double- vs. single-column layout, but also the way in which tables and figures are rendered and positioned.

Finally, printed papers might contain **additional**

[1] https://github.com/nlpAThits/MMAX2

[2] While PubMedCentral® is an obvious choice here, other resources with different full-text data formats exist and can also be used. All that needs to be modified is the conversion step (see Section 2.2).

Proceedings of the BioNLP 2021 workshop, pages 168–179
June 11, 2021.

content in headers or footers (like e.g. download timestamps). Also, while the references/bibliography section is an integral part of a printed paper and will be covered by OCR, in XML documents it is often structurally kept apart from the actual document text.

Given these challenges, attempting data extraction from document *images* if the documents are available in PDF or even full-text format may seem unreasonable. We see, however, the following useful applications:

1. Manual Database Curation As mentioned above, manual database curation requires the extraction, normalization, and database insertion of scientific content, often from *paper* documents. Given a paper document in which a human expert curator has manually marked a word or sequence of words for insertion into the database, having a *link* from these words to their electronic counterparts can eliminate or at least reduce error-prone and time-consuming steps like manual re-keying. Also, already existing annotations of the electronic full-text[3] would also be accessible and could be used to inform the curation decision or to supplement the database entry.

2. Automatic PDF Highlighting for Manual *Triage* Database curation candidate papers are identified by a process called document *triage* (Buchanan and Loizides, 2007; Hirschman et al., 2012) which, despite some attempts towards automation (e.g. Wang et al. (2020)), remains a mostly manual process. In a nut shell, triage normally involves querying a literature database (like PubMed[4]) for specific terms, skimming the list of search results, selecting and skim-reading some papers, and finally downloading and printing the PDF versions of the most promising ones for curation (Venkatesan et al., 2019). Here, the switch from *searching* in the electronic full-text (or abstract) to *printing* the PDF brings about a loss of information, because the terms that caused the paper to be retrieved will have to be located again in the print-out. A word-level alignment between the full-text and the PDF version would allow to create an enhanced version of the PDF with highlighted search term occurrences *before* printing.

3. Biomedical Expression OCR Current state-of-the-art OCR systems are very accurate at recognizing standard text using Latin script and baseline typography, but, as already mentioned, they are less reliable for more typographically complex expressions like chemical formulae. In order to develop specialized OCR systems for these types of expressions, ground-truth data is required in which image regions containing these expressions are labelled with the correct characters and their positional information (see also Section 5). If aligned documents are available, this type of data can easily be created at a large scale.

The remainder of this paper is structured as follows. In Section 2, we describe our data set and how it was converted into the shared XML format. Section 3 deals with the actual alignment procedure, including a description of the optional pre- and post-processing measures. In Section 4, we present experiments in which we evaluate the performance of the implemented procedure, including an ablation of the effects of the individual pre- and post-processing measures. Quantitative evaluation alone, however, does not convey a realistic idea of the actual usefulness of the procedure, which ultimately needs to be evaluated in the context of real applications including, but not limited to, database curation. Section 4.2, therefore, briefly presents examples of the alignment and highlighting detection functionality and the biomedical expression OCR use case mentioned above. Section 5 discusses relevant related work, and Section 6 summarizes and concludes the paper with some future work.

All the tools and libraries we use are freely available. In addition, our implementation can be found at https://github.com/nlpAThits/BioNLP2021.

2 Data

For the alignment of a paper document with its electronic full-text counterpart, what is minimally required is an image of every page of the document, and a full-text XML file of the same document. The document images can either be created by scanning or by directly converting the corresponding PDF into an image. The latter method will probably yield images of a better quality, because it completely avoids the physical printing and subsequent scanning step, while the output of the former method will be more realistic. We experiment with both types of images (see Section 2.1). We identify a document by its DOI, and refer to the different versions as DOI_{xml} (from the full-text XML), DOI_{conv}, and DOI_{scan}. Whenever

[3] Like https://europepmc.org/Annotations

[4] https://pubmed.ncbi.nlm.nih.gov/

a distinction between DOI_{conv} and DOI_{scan} is not required, we refer to these versions collectively as DOI_{ocr}.

Printable PDF documents and their associated `.nxml` files are readily available at PMC-OAI.[5] In our case, however, printed paper versions were already available, as we have access to a collection of more than 6.000 printed scientific papers (approx. 30.000 pages in total) that were created in the SABIO-RK[6] Biochemical Reaction Kinetics Database project (Wittig et al., 2017, 2018). These papers contain manual highlighter markup at different levels of granularity, including the word, line, and section level. Transferring this type of markup from printed paper to the electronic medium is one of the key applications of our alignment procedure. Our paper collection spans many publication years and venues. For our experiments, however, it was required that each document was freely available both as PubMedCentral® full-text XML and as PDF. While this leaves only a fraction of (currently) 68 papers, the data situation is still sufficient to demonstrate the feasibility of our procedure. Even more importantly, the procedure is unsupervised, i.e. it does not involve learning and does not require any training data.

2.1 Document Image to Multilevel XML

Since we want to compare downstream effects of input images of different quality, we created both a converted and a scanned image version for every document in our data set. For the DOI_{conv} version, we used `pdftocairo` to create a high-resolution (600 DPI) PNG file for every PDF page. Figure 1c shows an example. The DOI_{scan} versions, on the other hand, were extracted from 'sandwich' PDFs which had been created earlier by a professional scanning service provider. The choice of a service provider for this task was only motivated by the large number of pages to process, and not by expected quality or other considerations. A sandwich PDF contains, among other data, the document plain text (as recognized by the provider's OCR software) and a background image for each page. This background image is a by-product of the OCR process in which pixels that were recognized as parts of a character are inpainted, i.e. removed by being overwritten with colors of neighbouring regions. Figure 1b shows the background image corresponding to the page in Figure 1a. Note how the image retains the highlighting. We used `pdfimages` to extract the background images (72 DPI) from the sandwich PDF for use in highlighting extraction (see Section 2.1.1 below). We refer to these versions as DOI_{scan_bg}. For the actual DOI_{scan} versions, we again used `pdftocairo` to create a high-resolution (600 DPI) PNG file for every scanned page.

OCR was then performed on the DOI_{conv} and the DOI_{scan} versions with tesseract 4.1.1[7], using default recognition settings (`-oem 3 -psm 3`) and specifying hOCR[8] with character-level bounding boxes as output format. In order to maximize recognition accuracy (at the expense of processing speed), the default language models for English were replaced with optimized LSTM models[9]. No other modification or re-training of tesseract was performed. In a final step, the hOCR output from both image versions was converted into the MMAX2 (Müller and Strube, 2006) multilevel XML annotation format, using *words* as tokenization granularity, and storing word- and character-level confidence scores and bounding boxes as MMAX2 attributes.[10]

2.1.1 Highlighting Detection

Highlighting detection and subsequent extraction can be performed if the scanned paper documents contain manual markup. In its current state, the detection procedure described in the following *requires* inpainted OCR background images which, in our case, were produced by the third-party OCR software used by the scanning service provider. tesseract, on the other hand, does *not* produce these images. While it would be desirable to employ free software only, this fact does not severely limit the usefulness of our procedure, because 1) other software (either free or commercial) with the same functionality might exist, and 2) even for document collections of medium size, employing an external service provider might be the most economical solution even in academic / research settings, anyway. What is more, inpainted backgrounds are only required if highlighting detection is desired: For text-only alignment, plain scans are sufficient.

[5] https://www.ncbi.nlm.nih.gov/pmc/tools/oai/

[6] http://sabio.h-its.org/

[7] https://github.com/tesseract-ocr/tesseract

[8] http://kba.cloud/hocr-spec/1.2/

[9] https://github.com/tesseract-ocr/tessdata_best

[10] See the lower part of Figure A.1 in the Appendix.

(a) DOI$_{scan}$ (b) DOI$_{scan_bg}$ (c) DOI$_{conv}$

Figure 1: Three image renderings of the same document page: Scanned print-out w/ manual markup (a), background with markup only (b), and original PDF (c).

The actual highlighting extraction works as follows (see Müller et al. (2020) for details): Since document highlighting comes mostly in strong colors, which are characterized by large differences among their three component values in the RGB color model, we create a binarized version of each page by going over all pixels in the background image and setting each pixel to 1 if the pairwise differences between the R, G, and B components are above a certain threshold (50), and to 0 otherwise. This yields an image with regions of higher and lower density of black pixels. In the final step, we iterate over the word-level tokens created from the hOCR output and converted into MMAX2 format earlier, compute for each word its degree of highlighting as the percentage of black pixels in the word's bounding box, and store that percentage value as another MMAX2 attribute if it is at least 50%. An example result will be presented in Section 4.2.

2.2 PMC® `.nxml` to Multilevel XML

The `.nxml` format employed for PubMedCentral® full-text documents uses the JATS scheme[11] which supports a rich meta data model, only a fraction of which is of interest for the current task. In principle, however, all information contained in JATS-conformant documents can also be represented in the multilevel XML format of MMAX2. The `.nxml` data provides precise information about both the textual content (including correctly encoded special characters) and its word- and section-level layout. At present, we only extract content from the <**article-meta**> section (<article-title>, <surname>, <given-names>, <xref>, <email>, <aff>, and <abstract>), and from the <**body**> (<sec>, <p>, <tr>, <td>, <label>, <caption>, and <title>). These sections cover the entire textual content of the document. We also extract the formatting tags <italic>, <bold>, <underline>, and in particular <sup> and <sub>. The latter two play a crucial role in the chemical formulae and other domain-specific expressions. Converting the `.nxml` data to our MMAX2 format is straightforward.[12] In some cases, the `.nxml` files contain embedded LaTex code in <tex-math> tags. If this tag is encountered, its content is processed as follows: LaTex Math encodings for sub- and superscript, _{} and ^{}, are removed, their content is extracted and re-inserted with JATS-conformant _{...} and ^{...} elements. Then, the resulting LaTex-like string is sent through the `detex` tool to remove any other markup. While this obviously cannot handle layouts like e.g. fractions, it still preserves many simpler expressions that would otherwise be lost in the conversion.

[11] https://jats.nlm.nih.gov/archiving/

[12] See the upper part of Figure A.1 in the Appendix.

3 Outline of the Alignment Procedure

The actual word-level alignment of the DOI$_{xml}$ version with the DOI$_{ocr}$ version of a document operates on lists of $< token, id >$ tuples which are created from each version's MMAX2 annotation. These lists are characterized by longer and shorter stretches of tuples with matching tokens, which just happen to start and end at different list indices. These stretches are interrupted at times by (usually shorter) sequences of tuples with non-matching tokens, which mostly exist as the result of OCR errors (see below). *Larger* distances between stretches of tuples with matching tokens, on the other hand, can be caused by structural differences between the DOI$_{xml}$ and the DOI$_{ocr}$ version, which can reflect actual layout differences, but which can also result from OCR errors like incorrectly joining two adjacent lines from two columns.

The task of the alignment is to find the correct mapping on the token level for as many tuples as possible. We use the `align.globalxx` method from the `Bio.pairwise2` module of Biopython (Cock et al., 2009), which provides pairwise sequence alignment using a dynamic programming algorithm (Needleman and Wunsch, 1970). While this library supports the definition of custom similarity functions for minimizing the alignment cost, we use the most simple version which just applies a binary (=identity) matching scheme, i.e. full matches are scored as 1, all others as 0. This way, we keep full control of the alignment, and can identify and locally fix non-matching sequences during post-processing (cf. Section 3.2 below). The result of the alignment (after optional pre- and post-processing) is an n-to-m mapping between $< token, id >$ tuples from the DOI$_{xml}$ and the DOI$_{ocr}$ version of the same document.[13]

3.1 Pre-Processing

The main difference between pre- and post-processing is that the former operates on two still *unrelated* tuple lists of different lengths, while for the latter the tuple lists have the same length due to padding entries («GAP») that were inserted by the alignment algorithm in order to bridge sequences of non-alignable tokens. Pre-processing aims to smooth out trivial mismatches and thus to help alignment. Both pre- and post-processing, however, only modify the tokens in DOI$_{ocr}$, but never those in DOI$_{xml}$, which are considered as gold-standard.

[13] See also the central part of Figure A.1 in the Appendix.

Pre-compress matching sequences [`pre_compress=p`] The space complexity of the Needleman-Wunsch algorithm is $O(mn)$, where m and n are the numbers of tuples in each document. Given the length of some documents, the memory consumption of the alignment can quickly become critical. In order to reduce the number of tuples to be compared, we apply a simple pre-compression step which first identifies sequences of p tuples (we use $p = 20$ in all experiments) with *perfectly identical* tokens in both documents, and then replaces them with single tuples where the token and id part consist of concatenations of the individual tokens and ids. After the alignment, these compressed tuples are expanded again.

While pre-compression was always performed, the pre- and post-processing measures described in the following are optional, and their individual effects on the alignment will be evaluated in Section 4.1.

De-hyphenate DOI$_{ocr}$ tokens [`dehyp`] Sometimes, words in the DOI$_{ocr}$ versions are hyphenated due to layout requirements which, in principle, do not exist in the DOI$_{xml}$ versions. These words appear as three consecutive tuples with either the '-' or '¬' token in the center tuple. For de-hyphenation, we search the tokens in the tuple list for DOI$_{ocr}$ for single hyphen characters and reconstruct the potential un-hyphenated word by concatenating the tokens immediately before and after the hyphen. If this word exists *anywhere* in the list of DOI$_{xml}$ tokens, we simply substitute the three original $< token, id >$ DOI$_{ocr}$ tuples with one merged tuple. De-hyphenation (like all other pre- and post-processing measures) is completely lexicon-free, because the decision whether the unhyphenated word exists is only based on the content of the DOI$_{xml}$ document.

Diverging tokenizations in the DOI$_{xml}$ and DOI$_{ocr}$ document versions are a common cause of local mismatches. Assuming the tokenization in DOI$_{xml}$ to be correct, tokenizations can be fixed by either joining or splitting tokens in DOI$_{ocr}$.

Join incorrectly split DOI$_{ocr}$ tokens [`pre_join`] We apply a simple rule to detect and join tokens that were incorrectly split in DOI$_{ocr}$. We move a window of size 2 over the list of DOI$_{ocr}$ tuples and concatenate the two tokens. We then iterate over all tokens in the DOI$_{xml}$ version. If we find the reconstructed word in a matching context (one immediately

preceeding and following token), we replace, in the DOI$_{ocr}$ version, the first original tuple with the concatenated one, assigning the concatenated ID as new ID, and remove the second tuple from the list. Consider the following example.

```
< phen ,    word_3084 >n
< yl ,      word_3085 >n+1
⟹
< phenyl ,  word_3084+word_3085 >n
```

This process (and the following one) is repeated until no more modifications can be performed.

Split incorrectly joined DOI$_{ocr}$ tokens [`pre_split`] In a similar fashion, we identify and split incorrectly joined tokens. We move a window of size 2 over the list of DOI$_{xml}$ tuples, concatenate the two tokens, and try to locate a corresponding single token, in a matching context, in the list of DOI$_{ocr}$ tuples. If found, we replace the respective tuple in that list with two new tuples, one with the first token from the DOI$_{xml}$ tuple and one with the second one. Both tuples retain the ID from the original DOI$_{ocr}$ tuple. In the following example, the correct tokenization separates the trailing number 3 from the rest of the expression, because it needs to be typeset in subscript in order for the formula to be rendered correctly.

```
< KHSO3,    word_3228 >n
⟹
< KHSO,     word_3228 >n
< 3,        word_3228 >n+1
```

3.2 Alignment Post-Processing

Force-align [`post_force_align`] The most frequent post-processing involves cases where single tokens of the same length and occurring in the same context are not aligned automatically. In the following, the left column contains the DOI$_{ocr}$ and the right the DOI$_{xml}$ tuples. In the first example, the β was not correctly recognized and substituted with a B. We identify force-align candidates like these by looking for sequences of s consecutive tuples with a «GAP» token in one list, followed by a similar sequence of the same length in the other. Then, if both the context and the number of characters matches, we force-align the two sequences.

```
<metallo , word_853>     <metallo , word_546>
<-, word_854>            <-, word_547>
<B, word_855>            <<<GAP>>, ->
<<<GAP>>, ->             < β, word_548>
<-, word_856>            <-, word_549>
<lactamase , word_857>   <lactamase , word_550>
⟹
...
<B, word_855>            < β, word_548>
...
```

For $s = 2$, force-align will also fix the following.

```
<acid , word_1643>       <acid , word_997>
<,, word_1644>           <,, word_998>
<1t , word_1645>         <<<GAP>>, ->
<1s , word_1646>         <<<GAP>>, ->
<<<GAP>>, ->             <it , word_999>
<<<GAP>>, ->             <is , word_1000>
<purified , word_1647>   <purified , word_1001>
<using , word_1648>      <using , word_1002>
⟹
...
<1t , word_1645>         <it , word_999>
<1s , word_1646>         <is , word_1000>
...
```

4 Experiments

4.1 Quantitative Evaluation

We evaluate the system on our 68 DOI$_{xml}$ – DOI$_{ocr}$ document pair data set by computing P, R, and F for the task of aligning tokens from DOI$_{xml}$ (the gold-standard) to tokens in DOI$_{ocr}$. By defining the evaluation task in this manner, we take into account that the DOI$_{ocr}$ version usually contains more tokens, mostly because it includes the bibliography, which is generally *not* included in the DOI$_{xml}$ version. Thus, an alignment is perfect if every token in DOI$_{xml}$ is correctly aligned to a token in DOI$_{ocr}$, regardless of there being *additional* tokens in DOI$_{ocr}$. In order to compute P and R, the number of *correct* alignments (=TP) among all alignments needs to be determined. Rather than inspecting and checking all alignments manually, we employ a simple heuristic: Given a pair of automatically aligned tokens, we create two KWIC string representations, KWIC$_{xml}$ and KWIC$_{ocr}$, with a left and right context of 10 tokens each. Then, we compute the normalized Levenshtein similarity $lsim$ between each pair $ct1$ and $ct2$ of left and right contexts, respectively, as

$$1 - levdist(ct1, ct2)/max(len(ct1), len(ct2))$$

We count the alignment as correct (=TP) if $lsim$ of **both** the two left **and** the two right contexts is $>= .50$, and as incorrect (=FP) otherwise.[14] The number of missed alignments (=FN) can be computed by substracting the number of TP from the number of all tokens in DOI$_{xml}$. Based on these counts, we compute precision (P), recall (R), and F-score (F) in the standard way. Results are provided in Table 1. For each parameter setting (first column), there are two result columns with P, R, and F each. The column **DOI$_{xml}$ – DOI$_{conv}$** contains alignment results for which OCR was

[14] Figure A.2 in the Appendix provides an example.

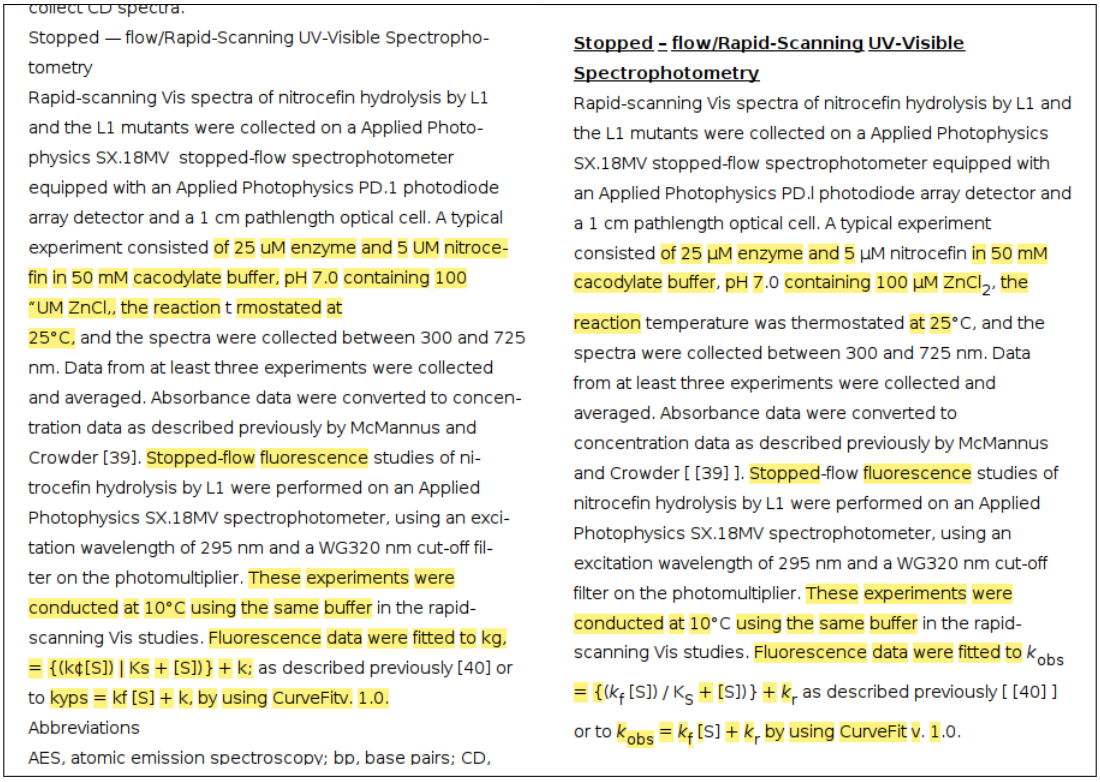
Pre-/Post-Processing	$\mathbf{DOI}_{xml}$ – $\mathbf{DOI}_{conv}$			$\mathbf{DOI}_{xml}$ – $\mathbf{DOI}_{scan}$		
	P	R	F	P	R	F
–	**95.04**	76.90	85.01	**93.59**	75.29	83.45
dehyp	94.91	77.47	85.31	93.48	75.96	83.81
pre	**95.04**	77.40	85.32	93.57	75.83	83.77
dehyp + pre	94.90	77.97	85.61	93.47	76.52	84.15
post_force_align	95.03	78.57	86.02	93.57	76.99	84.48
dehyp + post_force_align	94.91	79.17	86.32	93.47	77.69	84.86
pre + post_force_align	95.02	79.08	86.32	93.56	77.55	84.81
dehyp + pre + post_force_align	94.90	**79.68**	**86.63**	93.47	**78.27**	**85.20**

Table 1: Alignment Scores (micro-averaged, n=68). All results using pre_compress=20. Max. values in bold.

performed on the converted PDF pages, while results in column $\mathbf{DOI}_{xml}$ – $\mathbf{DOI}_{scan}$ are based on scanned print-outs. Differences between these two sets of results are due to the inferior quality of the images used in the latter. The top row in Table 1 contains the result of using only the alignment without any pre- or post-processing. Subsequent rows show results for all possible combinations of pre- and post-processing measures (cf. Section 3.1). Note that pre_split and pre_join are not evaluated separately and appear combined as pre. The first observation is that, for $\mathbf{DOI}_{xml}$ – $\mathbf{DOI}_{conv}$ and $\mathbf{DOI}_{xml}$ – $\mathbf{DOI}_{scan}$, precision is very high, with max. values of 95.04 and 93.59, respectively. This is a result of the rather strict alignment method which will align two tokens only if they are *identical* (rather than merely *similar*). At the same time, precision is very stable across experiments, i.e. indifferent to changes in pre- and post-processing. This is because, as described in Section 3.1, pre- and post-processing exclusively aim to improve recall by either smoothing out trivial mismatches before alignment, or adding missing alignments afterwards. In fact, pre- and post-processing actually *introduce* precision errors, since they relax this alignment condition somewhat: This is evident in the fact that the two top precision scores result from the setup with no pre- or post-processing at all, and even though the differences across experiments are extremly small, the pattern is still clear. Table 1 also shows the intended positive effect of the different pre- and post-processing measures on recall. Without going into much detail, we can state the following: For $\mathbf{DOI}_{xml}$ – $\mathbf{DOI}_{conv}$ and $\mathbf{DOI}_{xml}$ – $\mathbf{DOI}_{scan}$, the lowest recall results from the setup without pre- or post-processing. When pre- and post-processing measures are added, recall increases constantly, at the expense of small drops in precision. However, the positive effect consistently outweighs the negative, causing the F-score to increase to a max. score of 86.63 and 85.20, respectively, when all pre- and post-processing measures are used. Finally, as expected, the inferior quality of the data in DOI_{scan} as compared to DOI_{conv} is nicely reflected in consistently lower scores across all measurements. The absolute differences, however, are very small, amounting to only about 1.5 points. This might be taken to indicate that converted (rather than printed and scanned) PDF documents can be functionally equivalent as input for tasks like OCR ground-truth data generation.

4.2 Qualitative Evaluation and Examples

This section complements the quantitative evaluation with some illustrative examples. Figure 2 shows two screenshots in which DOI_{scan} (left) and DOI_{xml} (right) are displayed in the MMAX2 annotation tool. The left image shows that the off-the-shelf text recognition accuracy of tesseract is very good for standard text, but lacking, as expected, when it comes to recognising special characters and subscripts (like μ, $ZnCl_2$, or k_{obs} in the example). For the highlighting detection, the yellow text background was chosen as visualization in MMAX2 in order to mimick the physical highlighting of the printed paper. Note that since the highlighting detection is based on layout position only (and not anchored to text), manually highlighted text is recognized as highlighted regardless of whether the actual underlying text is recognized correctly. The right image shows the rendering of the correct text extracted from the original PMC® full-text XML. The rendering of the title as bold and underlined is based on typographic information that was extracted at conversion time (cf. Section 2.2). The same is true for the subscripts, which are correctly rendered both in terms of the content and the position. Table 2 displays a different type of result, i.e. a small selection of a much larger set of OCR errors with their respective images and the correct recognition result. This data, automatically identi-



collect CD spectra.

Stopped — flow/Rapid-Scanning UV-Visible Spectrophotometry

Rapid-scanning Vis spectra of nitrocefin hydrolysis by L1 and the L1 mutants were collected on a Applied Photophysics SX.18MV stopped-flow spectrophotometer equipped with an Applied Photophysics PD.1 photodiode array detector and a 1 cm pathlength optical cell. A typical experiment consisted of 25 uM enzyme and 5 UM nitrocefin in 50 mM cacodylate buffer, pH 7.0 containing 100 "UM ZnCl,, the reaction t rmostated at 25°C, and the spectra were collected between 300 and 725 nm. Data from at least three experiments were collected and averaged. Absorbance data were converted to concentration data as described previously by McMannus and Crowder [39]. Stopped-flow fluorescence studies of nitrocefin hydrolysis by L1 were performed on an Applied Photophysics SX.18MV spectrophotometer, using an excitation wavelength of 295 nm and a WG320 nm cut-off filter on the photomultiplier. These experiments were conducted at 10°C using the same buffer in the rapid-scanning Vis studies. Fluorescence data were fitted to kg, = {(k¢[S]) | Ks + [S])} + k; as described previously [40] or to kyps = kf [S] + k, by using CurveFitv. 1.0.

Abbreviations

AES, atomic emission spectroscopy; bp, base pairs; CD,

Stopped – flow/Rapid-Scanning UV-Visible Spectrophotometry

Rapid-scanning Vis spectra of nitrocefin hydrolysis by L1 and the L1 mutants were collected on a Applied Photophysics SX.18MV stopped-flow spectrophotometer equipped with an Applied Photophysics PD.l photodiode array detector and a 1 cm pathlength optical cell. A typical experiment consisted of 25 µM enzyme and 5 µM nitrocefin in 50 mM cacodylate buffer, pH 7.0 containing 100 µM $ZnCl_2$, the reaction temperature was thermostated at 25°C, and the spectra were collected between 300 and 725 nm. Data from at least three experiments were collected and averaged. Absorbance data were converted to concentration data as described previously by McMannus and Crowder [[39]]. Stopped-flow fluorescence studies of nitrocefin hydrolysis by L1 were performed on an Applied Photophysics SX.18MV spectrophotometer, using an excitation wavelength of 295 nm and a WG320 nm cut-off filter on the photomultiplier. These experiments were conducted at 10°C using the same buffer in the rapid-scanning Vis studies. Fluorescence data were fitted to k_{obs} = {(k_f [S]) / K_S + [S])} + k_r as described previously [[40]] or to k_{obs} = k_f [S] + k_r by using CurveFit v. 1.0.

Figure 2: DOI_{scan} document with automatically detected, overlayed highlighting (left). DOI_{xml} with highlighting transferred from automatically aligned DOI_{scan} document (right).

fied by the alignment post-processing, is a valuable resource for the development of biomedical expression OCR systems.

Image	OCR	PMC®
$\Delta\varepsilon_{265}$	Agyg5	$\Delta\varepsilon_{265}$
µM	“UM	μM
k_{obs}	kyps	k_{obs}
7 ± 1	7 + 1	7 ± 1
DH5α	DH50	DH5α

Table 2: Examples of image snippets (left) with incorrect (middle) and correct (right) text representation.

5 Related Work

The work in this paper is obviously related to automatic text alignment, with the difference that what is mostly done there is the alignment of texts in different languages (i.e. **bi-lingual alignment**). Gale and Church (1993) align not words but entire sentences from two languages based on statistical properties. Even if words were aligned, alignment candidates in bi-lingual corpora are not identified on the basis of simple matching, with the exception of language-independent tokens like e.g. proper names.

Scanning and OCR is also often applied to historical documents, which are only available in paper (Hill and Hengchen, 2019; van Strien et al., 2020; Schaefer and Neudecker, 2020). Here, **OCR post-correction** attempts to map words with word- and character-level OCR errors (similar to those found in our DOI_{ocr} data) to their correct variants, but it does so by using general language models and dictionaries, and not an aligned correct version. Many of the above approaches have in common that they employ specialized OCR models and often ML/DL models of considerable complexity.

The idea of using an electronic and a paper version of **the same document** for creating a character-level alignment dates back at least to Kanungo and Haralick (1999), who worked on OCR ground-truth data generation. Like most later methods, the procedure of Kanungo and Haralick (1999) works on the *graphical* level, as opposed to the *textual* level. Kanungo and Haralick (1999) use LaTex to cre-

ate what they call 'ideal document images' with controlled content. Print-outs of these images are created, which are then photocopied and scanned, yielding slightly noisy and skewed variants of the 'ideal' images. Then, corresponding feature points in both images are identified, and a projective transformation between these is computed. Finally, the actual ground-truth data is generated by applying this transformation for aligning the bounding boxes in the ideal images to their correspondences in the scanned images. Since Kanungo and Haralick (1999) have full control over the content of their 'ideal document images', extracting the ground-truth character data is trivial. The approach of van Beusekom et al. (2008) is similar to that of Kanungo and Haralick (1999), but the former use more sophisticated methods, including Canny edge detection (Canny, 1986) for finding corresponding sections in images of the original and the scanned document, and RAST (Breuel, 2001) for doing the actual alignment. Another difference is that van Beusekom et al. (2008) use pre-existing PDF documents as the source documents from which ground-truth data is to be extracted. Interestingly, however, their experiments only use synthetic ground-truth data from the UW3 data set[15], in which bounding boxes and the contained characters are explicitly encoded. In their conclusion, van Beusekom et al. (2008) concede that extracting ground-truth data from PDF is a non-trivial task in itself. Ahmed et al. (2016) work on automatic ground-truth data generation for *camera-captured* document images, which they claim pose different problems than document images created by scanning, like e.g. blur, perspective distortion, and varying lighting. Their procedure, however, is similar to that of van Beusekom et al. (2008). They also use pre-existing PDF documents and automatically rendered 300 DPI images of these documents.

6 Conclusions

In this paper, we described a completely unsupervised procedure for automatically aligning printed paper documents with their electronic full-text counterparts. Our point of departure and main motivation was the idea to alleviate the effect of the **paper-to-electronic media break** in manual biocuration, where printed paper is still very popular when it comes to close reading and manual markup. We also argued that the related task of document *triage* can benefit from the availability of alignments between electronic full-text documents (as retrieved from a literature database) and the corresponding PDF documents. Apart from this, we identified yet another field of application, biomedical expression OCR, which can benefit from ground-truth data which can automatically be generated with our procedure. Improvements in biomedical expression OCR, then, can feed back into the other use cases, by improving the OCR step and thus the alignment, thus potentially establishing a kind of bootstrapping development. Our implementation relies on *tried and tested* technology, including tesseract as off-the-shelf OCR component, Biopython for the alignment, and MMAX2 as visualization and data processing platform. The most computationally complex part is the actual sequence alignment with a dynamic programming algorithm from the Biopython library, which we keep tractable even for longer documents by using a simple pre-compression method. The main experimental finding of this paper is that our approach, although very simple, yields a level of performance that we consider suitable for practical applications. In quantitative terms, the procedure reaches a very good F-score of 86.63 on converted and 85.20 on printed and scanned PDF documents, with corresponding precision scores of 94.90 and 93.47, respectively. The negligible difference in results between the two types of images is interesting, as it seems to indicate that converted PDF documents, which are very easy to generate in large amounts, are almost equivalent to the more labour-intensive scans. In future work, we plan to implement solutions for the identified use cases, and to test them in actual biocuration settings. Also, we will start creating OCR ground-truth data at a larger scale, and apply that for the development of specialised tools for biomedical OCR. In the long run, procedures like the one presented in this paper might contribute to the development of systems that support curators to work in a more natural, practical, convenient, and efficient way.

Acknowledgements

This work was done as part of the project DeepCurate, which is funded by the German Federal Ministry of Education and Research (BMBF) (No. 031L0204) and the Klaus Tschira Foundation, Heidelberg, Germany. We thank the anonymous BioNLP reviewers for their helpful suggestions.

[15] http://tc11.cvc.uab.es/datasets/DFKI-TGT-2010_1

References

Sheraz Ahmed, Muhammad Imran Malik, Muhammad Zeshan Afzal, Koichi Kise, Masakazu Iwamura, Andreas Dengel, and Marcus Liwicki. 2016. A generic method for automatic ground truth generation of camera-captured documents. *CoRR*, abs/1605.01189.

Thomas M. Breuel. 2001. A practical, globally optimal algorithm for geometric matching under uncertainty. *Electron. Notes Theor. Comput. Sci.*, 46:188–202.

George Buchanan and Fernando Loizides. 2007. Investigating document triage on paper and electronic media. In *Research and Advanced Technology for Digital Libraries*, pages 416–427, Berlin, Heidelberg. Springer Berlin Heidelberg.

John F. Canny. 1986. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698.

Virginia Clinton. 2019. Reading from paper compared to screens: A systematic review and meta-analysis. *Journal of Research in Reading*, 42(2):288–325.

Peter J. A. Cock, Tiago Antao, Jeffrey T. Chang, Brad A. Chapman, Cymon J. Cox, Andrew Dalke, Iddo Friedberg, Thomas Hamelryck, Frank Kauff, Bartek Wilczynski, and Michiel J. L. de Hoon. 2009. Biopython: freely available python tools for computational molecular biology and bioinformatics. *Bioinform.*, 25(11):1422–1423.

William A. Gale and Kenneth W. Church. 1993. A program for aligning sentences in bilingual corpora. *Computational Linguistics*, 19(1):75–102.

Mark J. Hill and Simon Hengchen. 2019. Quantifying the impact of dirty OCR on historical text analysis: Eighteenth century collections online as a case study. *Digit. Scholarsh. Humanit.*, 34(4):825–843.

Lynette Hirschman, Gully Burns, Martin Krallinger, Cecilia Arighi, Kevin Cohen, Alfonso Valencia, Cathy Wu, Andrew Chatr-Aryamontri, Karen Dowell, Eva Huala, Anália Lourenco, Robert Nash, Anne-Lise Veuthey, Thomas Wiegers, and Andrew Winter. 2012. Text mining for the biocuration workflow. *Database : the journal of biological databases and curation*, 2012:bas020.

International Society for Biocuration. 2018. Biocuration: Distilling data into knowledge. *PLOS Biology*, 16(4):1–8.

Tapas Kanungo and Robert M. Haralick. 1999. An automatic closed-loop methodology for generating character groundtruth for scanned documents. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(2):179–183.

Maja Köpper, Susanne Mayr, and Axel Buchner. 2016. Reading from computer screen versus reading from paper: does it still make a difference? *Ergonomics*, 59(5):615–632. PMID: 26736059.

Christoph Müller and Michael Strube. 2006. Multi-level annotation of linguistic data with MMAX2. In Sabine Braun, Kurt Kohn, and Joybrato Mukherjee, editors, *Corpus Technology and Language Pedagogy: New Resources, New Tools, New Methods*, pages 197–214. Peter Lang, Frankfurt a.M., Germany.

Mark-Christoph Müller, Sucheta Ghosh, Maja Rey, Ulrike Wittig, Wolfgang Müller, and Michael Strube. 2020. Reconstructing manual information extraction with db-to-document backprojection: Experiments in the life science domain. In *Proceedings of the First Workshop on Scholarly Document Processing, SDP@EMNLP 2020, Online, November 19, 2020*, pages 81–90. Association for Computational Linguistics.

SB Needleman and CD Wunsch. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443—453.

Wataru Ohyama, Masakazu Suzuki, and Seiichi Uchida. 2019. Detecting mathematical expressions in scientific document images using a u-net trained on a diverse dataset. *IEEE Access*, 7:144030–144042.

Robin Schaefer and Clemens Neudecker. 2020. A two-step approach for automatic OCR post-correction. In *Proceedings of the The 4th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 52–57, Online. International Committee on Computational Linguistics.

Joost van Beusekom, Faisal Shafait, and Thomas M. Breuel. 2008. Automated OCR ground truth generation. In *The Eighth IAPR International Workshop on Document Analysis Systems, DAS 2008, September 16-19, 2008, Nara, Japan*, pages 111–117. IEEE Computer Society.

Daniel van Strien, Kaspar Beelen, Mariona Coll Ardanuy, Kasra Hosseini, Barbara McGillivray, and Giovanni Colavizza. 2020. Assessing the impact of OCR quality on downstream NLP tasks. In *Proceedings of the 12th International Conference on Agents and Artificial Intelligence, ICAART 2020, Volume 1, Valletta, Malta, February 22-24, 2020*, pages 484–496. SCITEPRESS.

A Venkatesan, N Karamanis, M Ide-Smith, J Hickford, and J McEntyre. 2019. Understanding life sciences data curation practices via user research [version 1; peer review: 1 approved, 1 approved with reservations]. *F1000Research*, 8(1622).

Jian Wang, Mengying Li, Qishuai Diao, Hongfei Lin, Zhihao Yang, and YiJia Zhang. 2020. Biomedical document triage using a hierarchical attention-based capsule network. *BMC Bioinformatics*, 21(380).

Ulrike Wittig, Maja Rey, Andreas Weidemann, Renate Kania, and Wolfgang Müller. 2018. SABIO-RK: an updated resource for manually curated biochemical reaction kinetics. *Nucleic Acids Research*, 46(D1):D656–D660.

Ulrike Wittig, Maja Rey, Andreas Weidemann, and Wolfgang Müller. 2017. Data management and data enrichment for systems biology projects. *Journal of biotechnology.*, 261:229–237.

Appendix

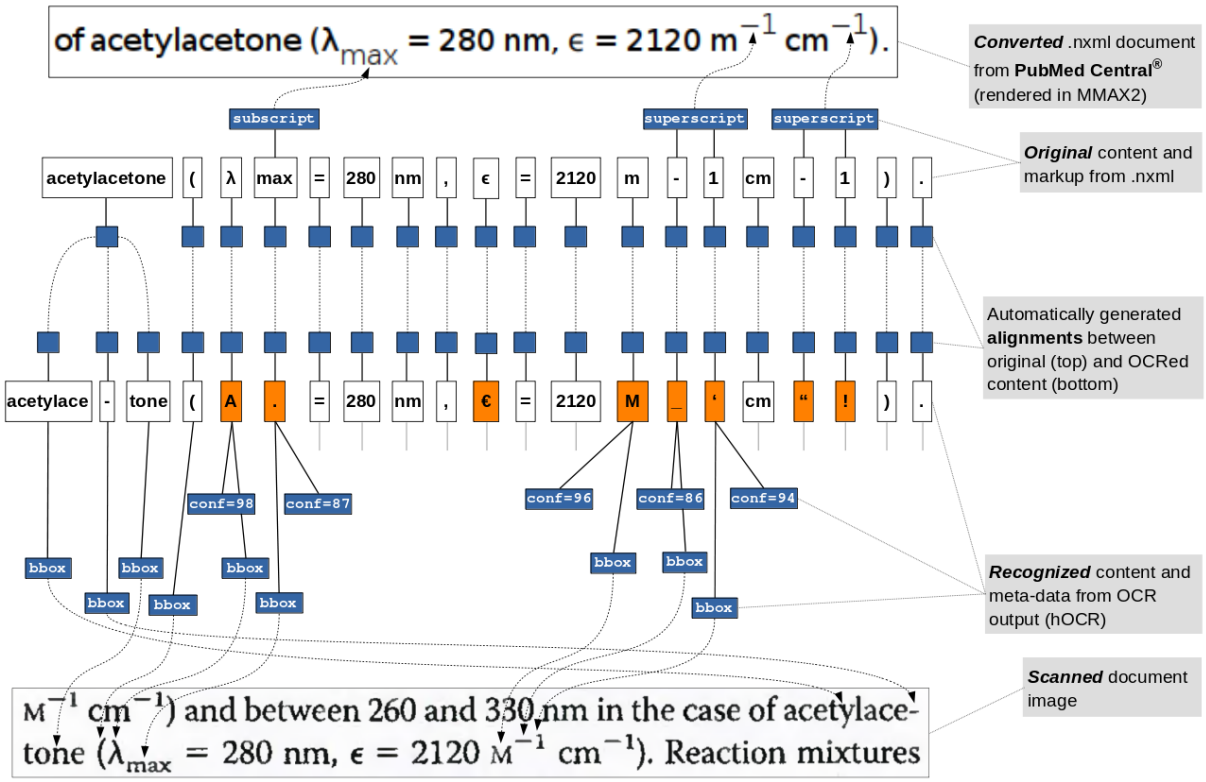


Figure A.1: Conversion and alignment data model. **Top**: Full-text and markup (`subscript`, `superscript`) is extracted from .nxml documents. Each content token is associated with an alignment token (solid blue boxes). **Bottom**: Text and meta-data is extracted from the OCR result of scanned document pages. Meta-data includes bounding boxes, which link the recognized text to image regions, and numerical recognition scores, which reflect the confidence with which the OCR system recognized the respective token. (Not all meta-data is given in the Figure to avoid clutter.)

```
                          MUT) DeficiencyFornyPatrick  1  2  † FroeseD >>  . <<    Sean  1  3  † SuormalaTerttu 1 YueWyatt W
                               .org 1,2% Communicated by Elizabeth F  >>  . <<    L Neufeld Received 6 June 2014; accepted revised manuscript
FP (0.136, 0.186)

                            Sean  1  3  † SuormalaTerttu 1 YueWyatt W  >>  . <<     3 BaumgartnerMatthias R.  1  21 Division for
                  6 June 2014; accepted revised manuscript 27 July 2014  >>  . <<    ABSTRACT: Methylmalonyl-CoA mutase (MUT) is
FP (0.208, 0.068)

  ZurichSwitzerland3 Structural Genomics ConsortiumUniversity of OxfordUK  >>  ABSTRACT << Methylmalonyl—CoA mutase (MUT) is
                  June 2014; accepted revised manuscript 27 July 2014.  >>  ABSTRACT << : Methylmalonyl-CoA mutase (MUT) is an
FP (0.169, 0.842)

Switzerland3 Structural Genomics ConsortiumUniversity of OxfordUKABSTRACT  >>  Methylmalonyl << —CoA mutase (MUT) is an
                ; accepted revised manuscript 27 July 2014. ABSTRACT:  >>  Methylmalonyl << -CoA mutase (MUT) is an essential enzyme
FP (0.247, 0.550)

                       University of OxfordUKABSTRACTMethylmalonyl—CoA >>  mutase << (MUT) is an essential enzyme in propionate catabolism
                 manuscript 27 July 2014. ABSTRACT: Methylmalonyl-CoA  >>  mutase << (MUT) is an essential enzyme in propionate catabolism
TP (0.538, 1.000)

                           of OxfordUKABSTRACTMethylmalonyl—CoA mutase >>  ( <<   MUT) is an essential enzyme in propionate catabolism that
                  27 July 2014. ABSTRACT: Methylmalonyl-CoA mutase  >>  ( <<   MUT) is an essential enzyme in propionate catabolism that
TP (0.667, 1.000)

                            OxfordUKABSTRACTMethylmalonyl—CoA mutase ( >>  MUT << ) is an essential enzyme in propionate catabolism that requires
                   July 2014. ABSTRACT: Methylmalonyl-CoA mutase (  >>  MUT << ) is an essential enzyme in propionate catabolism that requires
TP (0.702, 1.000)
```

Figure A.2: Sample output of the KWIC-based alignment evaluation procedure (context size = 10 tokens (left and right)). In each pair of lines, the top line comes from DOI_{ocr}, the bottom line from DOI_{xml}. Pairs are labeled as TP (correctly aligned) if the normalized Levenshtein similarity of both the left and the right context strings (given in parentheses) is above 0.5.