Survey of Post-OCR Processing Approaches

THI TUYET HAI NGUYEN, L3i, University of La Rochelle
ADAM JATOWT, University of Innsbruck
MICKAEL COUSTATY and ANTOINE DOUCET, L3i, University of La Rochelle

Optical character recognition (OCR) is one of the most popular techniques used for converting printed documents into machine-readable ones. While OCR engines can do well with modern text, their performance is unfortunately significantly reduced on historical materials. Additionally, many texts have already been processed by various out-of-date digitisation techniques. As a consequence, digitised texts are noisy and need to be post-corrected. This article clarifies the importance of enhancing quality of OCR results by studying their effects on information retrieval and natural language processing applications. We then define the post-OCR processing problem, illustrate its typical pipeline, and review the state-of-the-art post-OCR processing approaches. Evaluation metrics, accessible datasets, language resources, and useful toolkits are also reported. Furthermore, the work identifies the current trend and outlines some research directions of this field.

CCS Concepts: • **Computing methodologies** → *Natural language processing*; • **Applied computing** → *Document analysis*; *Text editing;*

Additional Key Words and Phrases: Post-OCR processing, OCR merging, error model, language model, machine learning, statistical and neural machine translation

ACM Reference format:
Thi Tuyet Hai Nguyen, Adam Jatowt, Mickael Coustaty, and Antoine Doucet. 2021. Survey of Post-OCR Processing Approaches. *ACM Comput. Surv.* 54, 6, Article 124 (July 2021), 37 pages.
https://doi.org/10.1145/3453476

1 INTRODUCTION

As the amount of born-analog documents is still quite large despite the recent wide shift to creating digital documents, substantial efforts have been devoted to transform paper-based materials into electronic texts for the purpose of text processing by computers (e.g., search or summarization), better preservation and easier access, often, by a much wider audience. The transformation process (i.e., digitisation) involves the efficient scanning or photographing of documents, page by page, and the conversion of the image of each page into computer-readable texts. The selection of digitisation techniques relies on several factors, such as the medium, printed vs. handwriting text, the language, and so on. The conversion of digital images into electronic texts is often realized

This work has been supported by the European Union's Horizon 2020 research and innovation programme under grant 770299 (NewsEye).
Authors' addresses: T. T. H. Nguyen, M. Coustaty, and A. Doucet, L3i, University of La Rochelle; emails: {hai.nguyen, mickael.coustaty, antoine.doucet}@univ-lr.fr; A. Jatowt, University of Innsbruck; email: adam.jatowt@uibk.ac.at.

0360-0300/2021/07-ART124 $15.00
https://doi.org/10.1145/3453476

by popular approaches: manual text entry, **optical character recognition (OCR)** software, and semi-automatic one.

Manual keying is naturally too expensive and may be associated with certain security issues when sharing information with a third party. According to the IMPACT project,[1] this method costs around 1 EUR per page, making the digitising price per book be around 400, 500 or even 1000 EUR. OCR software is a cost-efficient alternative of manual text entry without any associated security problems. This technique offers good recognition rates and has become one of the most popular and effective ways for the conversion of printed text. Along with manual text entry and OCR software, semi-automatic approaches allow to collaboratively transcribe paper-based documents into digital data that are then used to train OCR models for automatically generating transcripts. One of the computer-aided transcription tools is Transkribus [72][2] developed in the H2020 Project READ (Recognition and Enrichment of Archival Documents).[3]

Although OCR engines have been continuously improved and can already work well on modern text, they still lack adequate training data composed of past documents that is a strict requirement to achieve similarly high performance on historical texts. The physical quality of the original materials, complicated layouts, old fonts, and so on, all cause significant difficulties for the current OCR software. Consequently, OCR outputs are still noisy and possibly affect any downstream applications that use these textual materials as their input.

Many works have studied the impact of noisy input on **information retrieval (IR)** and **natural language processing (NLP)**. In IR, when digitised documents after OCR step are indexed, retrieval systems may miss some relevant documents in their responses to user queries and may return irrelevant ones. The report of van Strien et al. [158] confirms that the low quality of OCRed text[4] negatively affects information searching. Chiron et al. [26] estimate the impact of OCR errors on the Gallica digital library managed by the National Library of France. They indicate that 7% of common search terms, which are queried at least 35 times, are potentially affected by OCR errors. Information retrieval performance remains good for fairly high error rates on long texts [152], yet it drops dramatically [32, 105] on short texts. The results of Bazzo et al. [11] show that noticeable impacts start at a word error rate of 5% regardless of text length.

Regarding NLP, several applications, i.e., **named entity recognition (NER)**, **part-of-speech (POS)** tagging, text summarization, sentence boundary detection, topic modeling, sentiment analysis, text classification, named entity linking, and so on, are badly affected by OCR errors. Performance of NER tools, which locate proper names and categorise them into the set of predefined classes (i.e., person, location, organization), considerably degrades along with the increase in **error rate (ER)** of OCR output [59, 104, 158]. When the **word error rate (WER)** of the text increases from 0% to 2.7%, the F-score of the NER tool decreases around 3% [104]. With higher ER, the performance of NER drops much faster, for instance, from 90% to 60% when the WER of the text rises from 1% to 7% or when its **character error rate (CER)** increases from 8% to 20% [59].

The impact of OCR on other NLP tasks has also been measured in some research papers. POS tagging is the task of assigning the proper POS (e.g., noun, verb, pronoun, preposition, adverb, conjunction, etc.) to each word in the input text with respect to its lexical meaning and context. Xiaofan [88] and Mieskes et al. [101] reveal that the ER of the tagger increases linearly with that of OCR output. Starting from 5% CER, the POS taggers suffer significant degradation on both English and German data [101]. Text summarization, which creates a summary representing the

[1] http://www.impact-project.eu/about-the-project/concept/, accessed July 6, 2020.
[2] https://readcoop.eu/transkribus/, accessed August 5, 2020.
[3] https://read.transkribus.eu/, accessed August 5, 2020.
[4] The text that is generated by the OCR software is hereby called "OCRed text."

most salient content of the original text, tends to be also highly affected by noisy input even with slight increases in the noise level [70]. Furthermore, it has been evidenced that OCR errors lead to negative influences on sentence boundary detection [70, 90, 158], on topic modeling that discovers the abstract topics latent in an input collection of documents [108, 158], on sentiment analysis if sentiment bearing words have not been recognised correctly, on text classification [169], and on named entity linking that links named entities to external knowledge bases [125].

In overall, the quality of OCR output variously affects information retrieval as well as NLP tasks. Performance of applications designed and implemented on the assumption of clean data typically degrades on noisy texts. Consequently, it is important to produce cleaner OCR output.

Several projects (e.g., IMPACT, Europeana,[5] etc.) have embarked on the digitising large amounts of documents that constitute European cultural heritage. A large portion of this heritage remains to be transformed into digital form. Furthermore, a part of the historical text collections has been already processed by various OCR algorithms that have inferior performance to the current ones.

As a result, many digitised historical collections are still noisy and the development of OCR engines along with post-processing techniques is still in high demand. It is obvious that re-OCRing large corpora is time-consuming and costly. Therefore, on the one hand, researchers prioritize to analyse and improve existing data. In fact, enhancing post-processing models and analyzing the impact of OCR errors on downstream tasks are in the first recommendation in the agenda of digitising historical and multilingual textual resources of the research group of the Northeastern University [146]. However, they prepare standards for ground truth annotation, evaluation, and so on, and determine which collections need to be re-OCRed. The interest of the community in this field is also illustrated by the number of registrations (i.e., 69 teams in total) to the two recent competitions on post-OCR text correction organised in conjunction with the **International Conference on Document Analysis and Recognition (ICDAR)** in 2017 [25] and in 2019 [135].

The last survey on techniques of improving OCR results was published in 1997 [37]. Later, there were some reports about post-processing strategies specifically applied in the context of their projects (e.g., Reference [160]). It is then essential to provide a novel survey of the field that gives an overview on up-to-date post-processing approaches in general context and it is a main purpose of this work.

This survey can be beneficial for different groups of readers: non-experts can receive a global overview of post-OCR processing approaches; the researchers can use it as the basis for discussion, or even getting ideas for future research; and, finally, developers can consult the survey and choose the suitable technique, evaluation dataset, or toolkit for testing and developing their products. We make the following contributions in this work:

(1) The importance of improving quality of OCR output is clarified by reviewing the impact of noisy OCRed text on the downstream tasks. We define the post-OCR processing problem and illustrate the typical pipeline of methods aiming at post-processing OCRed results.
(2) We survey several post-processing approaches, categorise them into specific groups with respect to the human-dependence level, the extent of used information, and the applied models (e.g., language model, machine translation model, etc.), then, analyse their advantages as well as drawbacks.
(3) Popular evaluation metrics, accessible datasets are described. In addition, we report several open source code provided by prior post-OCR processing works, language resources, and useful toolkits.

[5] https://www.europeana.eu/en, accessed July 22, 2020.

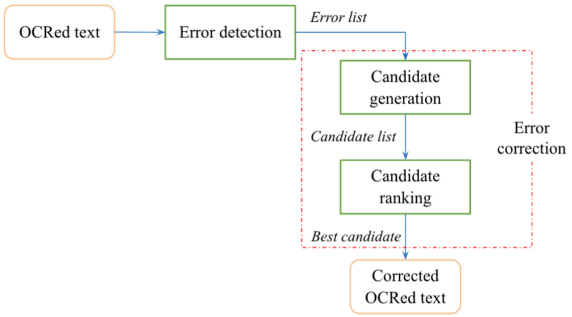


Fig. 1. Post-OCR processing pipeline.

(4) We identify the current trend of this field and outline research directions for the post-OCR processing task.

The rest of this article is structured as follows. In Section 3, we classify and discuss numerous post-OCR processing approaches. Section 4 presents common evaluation measures, accessible datasets, language resources and useful toolkits. Discussion points and potential future works are mentioned in Section 5. The survey concludes in Section 6.

2 DEFINITION OF POST-OCR PROCESSING PROBLEM

Given a sequence of n OCRed tokens $S = s_1 s_2 \dots s_n$, the objective of the post-OCR processing task is to find the true word sequence $W = w_1 w_2 \dots w_m$ that is printed in the original text.[6] Note that since resulting OCR sub-sequences may not be correct words due to segmentation errors, n is not always equal to m.

Most of approaches sequentially process each token of OCR output. Given an OCRed token s ($s \in S$), they need to select the actual word w ($w \in W$) that is recognised as s by the OCR software and is the most suitable one in the word context. If $s = w$, then s is a correct word, otherwise, s is an error. Considering the case when $s \neq w$, if $s \notin D$ (D as a dictionary), then s is called a *non-word* error; if $s \in D$, then s is known as a *real-word* error[7] that appears in the wrong context.

To choose w, probabilistic systems select w so that $P(w|s)$ and $P(w_1 w_2 \dots w_m)$ are highest. In terms of $P(w|s)$, after applying Bayes's rule and dropping the constant denominator $P(s)$, they can choose the word w that maximizes the following formula:

$$\hat{w} = \underset{w \in D}{\operatorname{argmax}} P(s|w) \times P(w), \tag{1}$$

where $\hat{w}$ is an estimate of the correct word w. The likelihood $P(s|w)$ is also called the error model, the prior $P(w)$ is the probability of w and is known as the language model.
In terms of $P(w_1 w_2 \dots w_m)$, it is known as the word ngram language model,

$$P(w_1 \dots w_m) = P(w_1) \times P(w_2|w_1) \times \dots \times P(w_n|w_{n-1} w_{n-2} \dots w_2 w_1). \tag{2}$$

Post-processing approaches consist of two tasks: *error detection* and *error correction*. Post-OCR error detection is for identifying incorrect tokens from the input text. The error detection supports human assistants to quickly correct errors, it also enables to flag noisy data for reprocessing them if necessary. Furthermore, the error detection produces a list of detected errors as the input of post-OCR error correction that is intended for rectifying invalid tokens. Given an error, typically,

[6]Images can be used as additional inputs for correcting OCR errors in some methods (e.g., References [35, 91, 93]).
[7]A *real-word* error is also called as a *false friend* one.

a list of word candidates is generated and scored based on available information. The highest-ranked candidate is then chosen for correcting the error in automatic correction approaches, or the top n candidates are suggested to the users in semi-automatic approaches. A typical post-OCR processing pipeline is illustrated in Figure 1.

3 POST-OCR PROCESSING TECHNIQUES

We first describe the procedure for collecting relevant papers. We searched papers containing "OCR post*," "OCR correct*," and "OCR detect*" in DBLP and Google Scholar in May 2020 and updated recent ones until December 2020. The resulting papers were carefully checked whether they present any post-processing methods or not. The relevant papers were then classified into main groups: manual, semi-automatic, and automatic ones. In the next step, we checked all papers that refer to or are cited by these selected-above papers and added the relevant works to our studied collection. We completed our collection with additional relevant papers that are chosen by going through DBLP of the authors of the selected papers. Our collection of papers in the field of post-processing OCRed text is categorised into subclasses that are discussed in detail in this section.

The literature of post-OCR processing research has a rich family of approaches. They can be grouped into three categories: manual, semi-automatic, and automatic types according to the human-dependence level. Nonetheless, there are a few semi-automatic approaches that are often combined with automatic ones, hence, we consider two main categories: manual and (semi-)automatic. The full categorization is represented as the following hierarchical tree. The characteristics of each group are discussed in the following sections. A brief description of each method and related information (e.g., performances, datasets, language resources, and toolkits) are also shown in a summary table that is provided as online supplementary material.

```
Post-OCR processing approaches
├── Manual approaches
└── (Semi-)automatic approaches
    ├── Isolated-word approaches
    │   ├── Merging OCR outputs
    │   ├── Lexical approaches
    │   ├── Error models
    │   ├── Topic-based language models
    │   └── Other models
    └── Context-dependent approaches
        ├── Language models
        │   ├── Statistical language models
        │   └── Neural network-based language models
        ├── Feature-based machine learning models
        └── Sequence-to-sequence (Seq2Seq) models
            ├── Traditional Seq2Seq models
            └── Neural network Seq2Seq models
```

3.1 Manual approaches

Crowd-sourcing approaches for post-processing of OCR output have been built to benefit from the public effort to enhance the quality of digitised texts.

One of the first crowd-sourcing approaches applied in post-OCR processing [65] is a web-based system called *Trove* in 2009. This system is developed by the National Library of Australia for correcting historical Australian newspapers. The approach presents full articles to volunteers, and allows fixing text line by line.

Clematide et al. [30] implement a crowd-correction platform called *Kokos* to reduce the error rate of the yearbooks of the Swiss Alpine Club digitised by Abby FineReader 7. This multilingual corpus contains texts of the 19th century written in German and French. *Kokos* shows full documents to users and lets them correct errors word by word. More than 180,000 characters on 21,247 pages were corrected by volunteers in about 7 months, with word accuracy of 99.7%.

Instead of showing full articles to users, another system (named *Digitalkoot*) developed by Chrons et al. [27] breaks the articles into single words without any context and puts them to simple games. The objective of the gaming model is to attract volunteers to donate their time for correcting erroneous tokens. It received relatively much attention with 4,768 people playing at least one game. The experimental result reveals that the quality of corrected text is very high with word accuracy over 99%. However, the system ignores nearby context and only allows users to interact with single words, which raises a doubt that it cannot correct *real-word* errors.

ReCAPTCHA is another type of public collaborative post-correction. **Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA)**, which is a widespread security measure on the World Wide Web, is used to limit the attack of malicious bots to websites. Von Ahn et al. [162] propose to harness the collective inputs from CAPTCHAs to digitise old printed material. They conceal crowd-sourcing post-correction effort behind an human authentication system to websites. Users are shown two images; the transcription of one is known to the system and used for verifying access to a website while the transcription of the other one is unknown and its content will be determined by a majority vote of contributors. Users do not know which one is known or unknown to the system. The authors report that the reCAPTCHA system achieves a word-level accuracy of 99.1%, whereas a standard OCR on the same set of articles obtains only 83.5%. Unfortunately, the version of reCAPTCHA allowing post-correction has been shutdown since March 2018.[8]

Collaborative post-OCR processing approaches prove their benefits with relatively high accuracy, and are cost effective. In addition, they can be easily applied to other digitisation projects. However, they heavily depend on volunteer work and it is necessary to keep the users motivated. Moreover, these approaches suffer from some risks caused by the public users, such as potential vandalism of text, incorrect suggestions. They also require the original documents that are often unavailable on some OCRed text corpora due to various restrictions such as copyrights or others (e.g., References [25, 47, 73, 135]).

3.2 (Semi-)automatic approaches

This approach type can be classified into isolated-word and context-dependent types based on the extent of information used in each approach. Isolated-word approaches only consider features of the target OCRed token itself, for example, its presence in a dictionary, the similarity between the token to a lexicon-entry, its frequency, recognition confidence score provided by the OCR software, and so on. Methods belonging to this kind mainly detect and fix *non-word* errors. However, context-dependent approaches not only take into account features of the OCRed token of focus but also its surrounding context, for instance, word ngram language model, parts of speech, and so on. By considering word context, approaches of this type are able to handle both *non-word* and *real-word* errors.

[8] https://developers.google.com/recaptcha/docs/versions#top_of_page, accessed July 14, 2020.

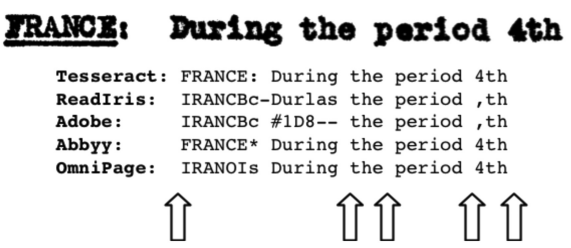


Fig. 2. A sample of aligned sequence hypotheses from five OCR engines of the same document. The arrows denote agreement points on white space among the aligned sequences. The hyphen "-" in a sequence denotes a gap aligned with characters in the other sequences [96].

3.2.1 Isolated-word approaches. Approaches of this type post-correct OCRed texts relying on characteristics of single tokens. Depending on the applied techniques, they can be classified into several groups, including methods that rely on merging OCR outputs, lexical approaches, error models, topic-based language models, and other models. The characteristics of each group are discussed in the following sections.

(1) Merging OCR outputs. One direction of work combines multiple OCR outputs, and typically includes three steps. OCRed texts are first created from multiple OCR engines operating on the same input or from the same OCR engine on different digital versions of the original document. OCRed texts can be also collected from text repetitions in the collection. Second, alignment algorithms are applied to align the OCR results. Last, different decision methods are explored to select the final output.

In the first step, the approaches of this type obtain multiple outputs by different ways. Whereas Lopresti and Zhou et al. [91] employ the same OCR engine on several scans of the same document, Lund et al. [92] adjust the binarization threshold to create multiple versions of the original documents. In contrast, Lin [87], Lund et al. [94–96], and Volk et al. [161] use multiple OCR engines to digitise the same document. Figure 2 illustrates a sample of aligned sequence hypotheses from five OCR engines.

Several alignment methods have been developed to align multiple OCR output sequences. Lund et al. [93] introduce an efficient algorithm to align the output of multiple OCR engines using the A* algorithm and then to utilize the differences between them. Instead of aligning OCR versions of the same scan, an approach of Wemhoener et al. [163] enables to create a sequence alignment of OCR outputs with the scans of different copies of the same book, or its different editions. Al Azawi et al. [4, 8] apply Line-to-Page alignment that aligns each line of the 1st OCR with the whole page of the second OCR using **Weighted Finite-State Transducers (WFST)**.

In the last step, several techniques are applied to choose the best sequence. Lopresti et al. [91], Lin [87], Wemhoener et al. [163], and Reul et al. [129] utilize voting policy, Al Azawi et al. [4, 8] use **Long Short-Term Memory (LSTM)** [64] to decide the most relevant output. Different kinds of features (voting, number, dictionary, gazetteer, and lexical feature) are used in learning decision list, maximum entropy classification or **conditional random fields (CRF)** methods to choose the best possible correction by Lund et al. [92, 94–96].

The above-mentioned approaches tend to be characterized by a lower word error rate than ones obtained when using a single OCR engine. However, most of them limit candidate suggestions to just the recognition results of OCR engines. In addition, these approaches do not consider any contextual information, thus, *real-word* errors cannot be corrected. Furthermore, they require some additional efforts of multiple OCR processing and the presence of the original OCR input that is not always available.

To alleviate the disadvantage of performing OCR many times, some recent methods benefit from text repetitions or k models of k-fold cross-validation. A work of Xu and Smith [167] identifies repeated texts in the collection and builds a multiple sequence alignment from them. The best sequence is chosen by majority voting or character ngram language model. Das et al. [35] also make use of duplicate words in the corpus. Words are grouped based on the similarity of images and text features. Each instance of the word groups is checked for its correctness via a dictionary. A word group containing errors accepts the most frequent prediction or the human input as the representative of the whole group. The approach of Reul et al. [129] benefits from k models of k-fold cross-validation and the majority voting. In particular, rather than choosing the best model from k-fold cross-validation and applying only this model on the test data, the authors examine all of the k models on the test data and choose the highest-voting sequence as the final output.

(2) Lexical approaches. Another line of work is the lexical approaches that typically rely on lexicons (or word unigram language model) and distance metrics for selecting candidates of OCR errors. The coverage of a dictionary, the way of constructing dynamic dictionary or utilizing available indexed data, different distance measures as well as complement features are discussed below.

Several popular measures are developed, such as the Damerau-Levenshtein edit distance [33, 85], which is the minimum number of single-character edit operations or ngram distance [7] that depends on the number of common ngrams between two strings. Schulz et al. [142] suggest some refinements of the Levenshtein edit distance (denoted as LV distance) based on frequent error patterns. Their method efficiently selects small candidate sets with higher recall, especially in the context of large lexicons.

Some typical examples of this type are Taghva et al. [153], Estrella et al. [46], Kettunen et al. [74], Cappelatti et al. [22]. Taghva et al. [153] introduce the document processing system named MANICURE, which includes a post-process module—PPSYS. This module detects errors mainly depending on dictionaries and corrects them by approximation matching using word frequency and character confusion. Estrella et al. [46] and Kettunen et al. [74] detect spurious tokens relying on dictionary, and word unigram frequency. For each invalid token, a list of alternatives is generated based on edit distance, and typographical characteristics. Afterwards, the most frequent candidate replaces the token. Cappelatti et al. [22] apply a spelling checker to suggest correction candidates and rank them by using LV distance or modified Needleman-Wunsch that considers frequency of the letter obtained from the corpus.

Considering the fact that conventional dictionaries are short of a considerable number of words of a specific domain, some prior works aim to study the influence of lexical coverage on post-OCR approaches and suggest techniques to dynamically build domain-dependent lexicons.

Strohmaier et al. [150] exploit thematic dictionary to improve results of approaches belonging to the lexical type. They build a dynamic dictionary from collecting vocabularies of Web pages of the input domain, which obtains a higher coverage than a static conventional dictionary. Some dynamic dictionaries created in this way are applied in their next work [149] in which they develop a tool for optimising lexical post-correction. Their tool includes two phases, i.e., (1) selecting parameters for optimising a post-correction model applied on a single OCR engine; (2) combining results of multiple OCR engines. The article mostly focuses on the first one. Given a set of dictionaries D, for each token of the OCR output, a list of n candidates is selected from a dictionary D_i ($D_i \in D$) based on an edit distance d_i and word frequency. The tool can help to choose the relevant values for n, D_i, d_i.

Similarly, Mihov et al. [102] automatically create dynamic dictionaries of domain-specific documents via the analysis of relevant web pages. In their approach, a dictionary is compiled as deterministic finite-state automaton (i.e., LV automata) to suggest correction candidates. They obtain a better lexical coverage than the previous work [150] by using vocabulary of the 1000 top-ranked

pages that are returned by a search engine with the input as 25 non-function words of the given OCR-corpus.

Whereas several studies concentrate on constructing special dictionaries, Gotscharek et al. [55] introduce some guidelines to design and select optimized lexical resources of distinct periods. Relying on their experiments on historical German collections, they emphasize the helpfulness of the dictionary created from words in the **ground-truth (GT)** text.

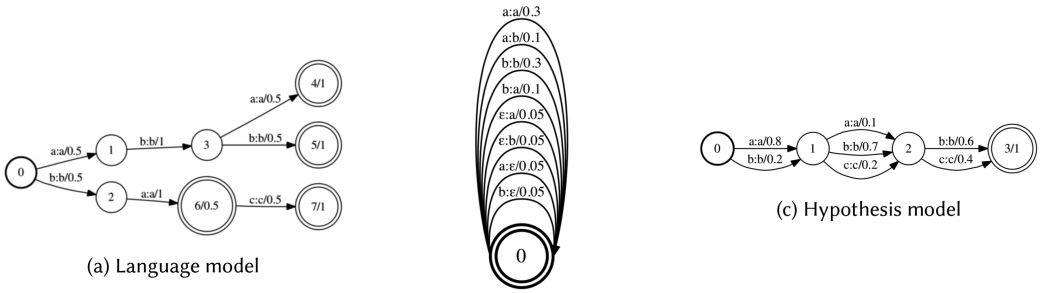
Instead of constructing a dictionary, Bassil et al. [10], Taghva and Agarwal [151] harness the Google's massive indexed data for post-processing OCR output. They send OCRed tokens to Google search engine as search queries. If the query contains errors, then the search engine will suggest some replaceable words for misspellings. These suggestions are used as corrections for OCR errors. Both of these methods obtain high performances on small datasets. Taghva and Agarwal [151] additionally use LV distance, longest common subsequence, the OCR confusion matrix to select a adequate candidate. Their method is able to correct more errors than using only Google's suggestions with about 16.6% improvement. Moreover, their analysis shows that Google search engine fails to suggest candidates if the context nearby the error is incorrect. However, it should be noted that their examples to illustrate this argument are actually not valid anymore after having checked the current output from the Google search engine.

Besides lexicons and distance metrics, some approaches of this type make use of complementary information to improve their performances. Furrer et al. [50] identify OCRed errors originating from Gothic texts by various resources, e.g., a large German dictionary, a list of local place names, the recognition confidence. Candidates are generated by similar character substitutions or character ngram distances. Their experiment on 35,000 words shows a word accuracy increase from 96.72% to 98.36%. Along with edit distance and word frequency, Hammarström et al. [60] and Jean-Caurant et al. [68] consider one more feature: distributional similarity (e.g., Word2Vec [103] or Glove [120]) to identify possible spelling variants of each word in GT text. Then, if a variant occurs in OCRed text, then it will be replaced by its corresponding correct word.

Reynaert [131, 132] introduces an unsupervised method to solve problems of lexical variations at word level [131] or character level [132]. The author relies on large background corpora to compute word frequencies and to create a lexicon if it is not available. The method exploits a hash table and a hash function to produce a large number for identifying anagrams that have the same characters (e.g., "example" and "eaxmple" have the same hash value). This feature enables to retrieve similar words for a given (or target) word by addition, subtraction or both on its hash value. For example, subtracting the hash value of the character "y" from the word "they" will result in the new word "the." Once retrieving all variants of a given word, the edit distance and word frequency are applied to choose the best-matching candidate. Reynaert [132] shows that the character-level approach works faster than their corresponding word-level approach on both of their evaluation corpora.

Lexical approaches are easy to apply, however, they have certain shortcomings. The lack of high-coverage lexicons is the most difficult problem of this type [102], especially with historical collections whose texts do not follow standard spellings. Moreover, the approaches belonging to this type only concentrate on single words, therefore, they cannot handle *real-word* errors.

(3) Error models. Several post-OCR processing approaches concentrate on error and/or character language models to fix erroneous OCRed strings. Early error models often rely on LV distance measures that weight all edits equally and ignore context. In the context of spelling correction, Church and Gale [28] develop an error model associating probabilities with single-character transformation. Insertion and deletion probabilities are conditioned on the preceding character. Brill and Moore [19] propose a more general error model that allows multiple-character edit operations along with their probabilities. These techniques are also applied to correct OCR errors.



(a) Language model

(b) Error model

(c) Hypothesis model

Fig. 3. Example of language model (a), error model (b), hypothesis model (c) in Reference [89]. (a): the probabilistic identity transducer with the language sample S={'aba', 'abb', 'ba', 'bac'}; (b): given two symbols s_1, s_2 and the empty symbol ε, three edit operations including substitutions, insertions and deletions are denoted as s_1:s_2, ε:s_2 and s_1:ε, respectively; (c): given alphabet ['a', 'b', 'c'], the confidence probabilities of the OCR output as [0.8, 0.2, 0.0], [0.1, 0.7, 0.2], [0.0, 0.6, 0.4]. In (c): considering the edges between node (0) and node (1), these probabilities mean that the first symbol of the OCR output is 'a' with probability 0.8, 'b' with probability 0.2, 'c' with probability 0.0 (transitions with probability 0.0 are not shown); the other edges work similarly.

Utilizing multiple-character edit operations, Taghva and Stofsky [154] implement an interactive post-processing method named OCRSpell. First, the method identifies the largest possible word boundary instead of simply tokenizing texts based on white-spaces. Each tokenized word is then checked for its correctness via a lexicon-lookup. Next, replaceable candidates for each erroneous token are generated based on the confusion matrix and on its variant that is updated when users add a correction for an error. In addition, a heuristic is designed to create candidates for words containing unrecognized character (i.e., a tilde ˜ in their paper). The Bayesian function is then employed to score suggested candidates based on collocation frequencies and character ngrams. Finally, the ranked list of suggestions is recommended to users.

Kolak and Resnik [82] focus on the noisy channel model and adapt a framework of syntactic pattern recognition to solve the problem of post-OCR processing task. Their probabilistic generative model is defined as three probability distributions, i.e., G, $Q(a)$, $S(b|a)$. G governs insertions in the transformation process. $Q(a)$ and $S(b|a)$ control the probability that symbol a in the input string will be inserted, or transformed into symbol b in the output string, respectively. They introduce practical techniques to estimate model parameters based on edit distance and IBM translation models (called TME). Performance of their TME error model is comparable to multiple-character edits [19] on a small training data, but, underperforms on a larger one.

In contrast, Perez-Cortes et al. [122] propose stochastic error correcting parsing for post-OCR processing. Their idea is to build a stochastic finite-state machine that accepts the strings in the lexicon. If a word is accepted by the model, then no correction is needed. Otherwise, an error model compiled in a finite-state machine is applied to suggest and rank candidates.

Extending the approach of Perez-Cortes et al. [122], Llobet et al. [89] build a character language model, an error model, and then add one more model built from character recognition confidences called a hypothesis model. Examples of these models are shown in Figure 3. Three models are compiled separately into WFSTs, then are composed into the final transducer. The best candidate is chosen by the lowest cost path of this final transducer. However, character recognition confidence is often unavailable in some digitised corpora, such as datasets used in these works [25, 47, 107, 135], so it is impossible to fully implement and apply the hypothesis model. This method outperforms the one of Perez-Cortes et al. [122] on correcting Spanish names at character level.

In another work, Perez-Cortes et al. [123] build the combined language models from related fields based on WFST. Three fields including provinces, municipalities and postal codes of Spain are conducted in their models. The experimental results show that they significantly reduce the error rates with acceptable correction time. The method is further improved with adaptive threshold to reject the less reliable hypotheses [110, 111].

Similarly, two post-OCR processing methods based on **Hidden Markov Model (HMM)** are employed by Borovikov et al. [17] and Richter et al. [134]. In these models, the hidden states are characters of words in a dictionary, the observable symbols are OCRed characters. The transition probability and the initial probability are computed from normalized bigram and initial character statistics from the dictionary and the ground truth. The emission probability is calculated from a confusion matrix.

Weighted Finite-State Machines (WFSM) technique is again applied in Kolak and Resnik [83] where the authors employ the noisy channel model using WFSM to estimate the most probable source character sequence for a given observed character sequence (or OCRed text). Their method allows both single-character (like in Reference [82]) or multiple-character edits (Reference [19]) and can work without a lexicon resource.

Kettunen et al. [75] present three approaches for improving the quality of Finnish digitised texts relying on corpus statistics, and edit distance. In their first two methods, they compute the "Finnishness" of an OCRed token (called "F") as a product of its character trigram frequencies, and compare "F" with a threshold to detect an error. The best-matching word is selected based on edit distance and word frequency. In the last one, the candidates are chosen based on WFST.

Similarly, Al Azawi et al. [3] post-correct OCRed text relying on error model and language model built in WFST. The novel thing is that they apply context-dependent confusion rules that take into account two characters on the leftmost and rightmost sides of the erroneous character. Silfverberg et al. [145] compile the probabilistic contextual error model into WFST for suggesting correction candidates for each input string. Weights used in WFST are extracted in two ways. In the first way, they are approximated by counting occurrences of output letters if input letters match the same context. In the second one, they are estimated by the perceptron tagger whose feature set considers relations between output letters, or between output and input letters. The method is applied to post-correct OCRed errors in the works of Drobac et al. [42, 43].

Reffle and Ringlstetter [127] argue that information on spelling variations and OCR errors plays a very important role in improving OCR output (e.g., fine-tuning post-OCR approaches) as well as in retrieving information over digitised historical texts. As a result, they suggest an unsupervised method to automatically analyse OCRed historical documents for profiling such information. The profile contains global and local information. The local profile provides a ranked list of possible correction suggestions for each OCR token by using the ground truth and historical spelling variations. Accumulating the local profile, they compute the global profile for an OCRed historical text. This global profile contains list of OCR error types and list of historical patterns with their estimated frequencies. Their results reveal a strong correlation between the actual information in the OCRed text and the one in their estimated profile. This profile can be applied to post-process digitised texts of the same language or being processed by the same OCR software.

This automated profiling mechanism is the most important part of an interactive OCR post-processing tool, PoCoTo [159]. Each OCR word with the corresponding image snippet and a full view of the page are shown in parallel to users. Utilising the global and local profiles, this tool highlights possible erroneous tokens of the OCR input text, computes, and suggests correction candidates to users. The tool also allows batch correction of error series.

In the revisited version of this method, Fink et al. [49] additionally refine profiles from user feedback of manual correction steps. They also enlarge their set of patterns with addition ones found

in documents of earlier periods. In addition, uninterpretable tokens are inserted to the error set, which helps to improve the recall of error detection. The profile is again utilized in a recent work of Englmeier et al. [45]. They introduce the A-PoCoTo system for fully automated post-processing in which the profile output information is used as one of the features for machine learning. A combined system between the A-PoCoTo and the interactive one, called as A-I-PoCoTo, allows users to fix incorrect results of automated correction, or to validate correct decisions.

In contrast to other approaches of this type, Gerdjikov et al. [53] suggest a novel method called **Regularities Based Embedding of Language Structures (REBELS)** for generating and ranking candidates based on word structures. The general idea is to see how noisy words and referenced words (noisy words as errors, referenced words as words in a dictionary) are constructed in terms of distinctive infixes. A distinctive infix is either a word prefix or a word infix that occurs in at least two distinct contexts. They assume that there is a set of spelling variations that transforms distinctive infixes of noisy words into those of their corresponding referenced words in a way that best matches with the structures of both sets. In a learning stage, REBELS processes a set of instances $I = (E_i, N_i)$ (in which $E_i \in H$, H as a list of noisy words, $N_i \in D$, D as a list of referenced words) and tries to decide a set of spelling variations P. In a testing stage, given a noisy word, REBELS decomposes it into distinctive infixes relying on the structure of the observed noisy words. Based on the decomposition, the spelling variations of P are applied to its different parts. Last, the A* algorithm is used to generate and rank their candidates.

(4) Topic-based language models. Several mentioned-above post-OCR approaches improve the quality of OCR output by combining error model and word unigram language model. However, the used language is global and independent of the document topic. In other words, topic information of the document is not considered to eliminate noisy candidates. Some studies suggest then to apply topic-based language models for post-processing OCRed texts.

Given an OCRed document and its list of errors, Wick et al. [165] and Bhardwaj et al. [13] suggest candidates by collecting lexicon-entries that have less than three characters differing from a given error. Then, these candidates are ranked relying on Equation (3),

$$Score(w_c) = P(w_c) \times \prod_{j}^{N} P\left(l_j^f | l_j^s\right); \; P(w_c) = \sum_{k}^{M} P(w_c|t_k) \times P(t_k), \tag{3}$$

where w_c is a candidate, N is the candidate length, $P(l_j^f | l_j^s)$ is the probability that character l_j^s being misrecognized as l_j^f, $P(w_c)$ is the probability of w_c, M is the number of topics in the model, $P(t_k)$ is the probability of the tested document belonging to the topic t_k. $P(t_k)$ is computed based on Latent Dirichlet Allocation topic model by Wick et al. [165], or Maximum Entropy model by Bhardwaj et al. [13].

(5) Other models. Apart from discussed-above models, there are some approaches relying on other different models, such as feature-based machine learning, string-to-string transformation, and **neural machine translation (NMT)**.

Dannélls and Persson [34] detect OCR errors by a **support vector machine (SVM)** classifier based on seven features: number of non-alphanumeric characters in a word, number of vowels in a word, number of upper case characters, tri-gram character frequencies, the presence of digit in a word, word frequency, and word length. Afterwards, they search lexicon-entries whose LV distances to detected errors are less than a threshold and use them as correction alternatives. The candidates are ranked by the edit distance and word frequency. Their post-processing method is applied on OCRed texts generated from three OCR engines, i.e., Abbyy Finereader, Tesseract, and Ocropus. They indicate challenges in developing a post-processing approach regardless of the OCR

systems. In fact, they only have some improvements in texts generated by Tesseract and perform best on words of length of four characters.

Eger et al. [44] examine four string-to-string transformation models in the paradigm of error correction: (1) Sequitur [14] is a joint sequence model that uses ngram probabilities over pairs of substrings of the input and output sequences; (2) DirectTL+ [69] views the task as a pipeline of sequence segmentation and sequence labeling. DirecTL+ integrates joint character ngrams in the sequence labeling; (3) AliSeTra [44] works in a similar way to DirectTL+. It uses CRF (i.e., conditional random field) as its sequence labeler and ignores joint character ngrams; (4) Contextual Edit Distance [31] regards the task as weighted edit operations that can be conditioned on input and output context. Three of these models (Sequitur, DirecTL+, AliSeTra) outperform the baselines, the best one achieves word accuracy around 88.35% while the baseline (i.e., the noisy model allowing multiple-character edits [19]) gets about 84.20%. The simple combination (e.g., majority voting) of these models and the baselines even obtains better results than the single models.

Hämäläinen and Hengchen [58] employ a character-level NMT model for rectifying errors on the 18th-century documents. To create training data, edit distance and Word2Vec trained on noisy data are employed. For each correct word, they query its most similar semantic words using Word2Vec. The resulting words are then categorized as correct words or errors. Next, they group each error with the most similar correct word in the resulting word list relying on edit distance and remove errors whose edit distances to their respective correct words are greater than three. Finally, they trained the NMT model on obtained parallel data using OpenNMT [79]. The top 10 candidates generated by the trained model are checked against a dictionary to choose the correction form of the given error.

In a similar way, Hakala et al. [57] train character-level NMT models using OpenNMT [79] and apply it to transform OCR errors into their respective correct words. However, they create training data relying on text duplication in the OCRed text collection. In particular, they search pairs of repeated texts in the corpus and cluster them. For each cluster of more than 20 sequences, they tokenize its sequences and group similar words based on LV distance. Each word group is then aligned and the most common character for each position is chosen as its representative. The replacement occurrence is computed for every character and the distributions are collected from frequencies of such replacements from all clusters. Lastly, they generate noisy strings by randomly deleting, inserting, and substituting one or two characters for a given string.

3.2.2 Context-dependent approaches. Post-OCR processing approaches of this kind handle not only *non-word* but also *real-word* errors by considering characteristics of single tokens and of their surrounding contexts. They are categorised into some following groups: language models, feature-based machine learning models, and sequence-to-sequence models.

(1) Language models. Several post-processing methods exploit language models for generating and ranking correction candidates. We divide them into two main groups according to the used language models, i.e., statistical and neural network-based language models.

(1a) Statistical language models. Statistical language models estimate the probability distribution of word sequences, which are derived from frequency counts with some smoothing techniques for handling data sparsity problems. For simplicity, the methods utilising word ngram frequency are also examined in this section.

Emphasizing the importance of high coverage and domain-sensitive dictionary in lexical approaches, Ringlstetter et al. [136, 137] refine the crawl strategy employed in the previous approach [150] to produce smaller dictionaries with high coverage. In particular, the similarity between crawled pages and the given input text is controlled based on the normalised cosine distance. For each token in the input, they select lexical words whose LV distances to the input

token are lower than a threshold. Correction suggestions are additionally ranked by word ngram frequencies from the crawled data.

Poncelas et al. [124], Hládek et al. [63], and Généreux et al. [52] employ similar techniques to detect errors and suggest correction candidates. Particularly, they detect noisy tokens by a lexicon-lookup and select candidates based on LV distances between a given error and lexicon-entries. However, they rank correction suggestions in different ways. Poncelas et al. [124] rank the correction suggestions based on word 5-gram language model built from Europarl-v9 corpus.[9] Hládek et al. [63] use HMM with state transition probability as word bigram language model probability and observation probability as their smoothing string distance for choosing the best candidate. Généreux et al. [52] choose the most probable candidate by a sum of the following feature values: confusion weight, candidate frequency, and bigram frequency. They report that their method performs comparably to the work of Hauser [61], which applies multiple-character edits. Both of these methods obtain similar error reduction rates on the datasets that have the same publication period and the same font.

Reffle et al. [126] show that error profiles combined with word trigram frequencies are useful to detect and correct *real-word* errors. The error profiles are produced based on a set of non-lexical strings (called E). Each entry of E is represented as the form (w, w', op) in which a noisy token (w) is created by applying only one edit operation (op) at a time to each word (w') of the lexicon D. For each token t of the input text T, if t appears in the lexicon D, then a triple (t, t, id) is added to L, where id denotes identity; all entries in E that have t as the first component are added to L. The error profile of T is the relative occurrence frequencies of edit operations in L. They then apply the frequent edit operations of T on lexicon entries and save all transformations (w, w', op) that lead to a correct word. If w occurs in the text, then w is possible a *real-word* error. Retrieving from the list of transformations of w, they suggest corresponding w' as possible correction candidates. The best relevant one is decided by word trigram frequencies.

Similarly, Evershed et al. [47] exploit both error and word ngram language models in their post-processing method. They carefully generate candidates at character level using the error model and at word level using "gap trigrams," which gives the probability of a word conditioned on its left and right neighbours. The error model uses weighted multiple character edits and an estimation relying on "reverse OCR" procedure that generates low resolution glyph bitmaps for a pair of noisy and correct words, then computes a bit correlation. Suggestions are ranked by confusion weight and word trigram plus 5-gram language model.

The competition team from Centro de Estudios de la RAE (denoted as WFST-PostOCR in the ICDAR2017 competition [25], RAE in the ICDAR2019 one [135]) compile probabilistic character error models into WFST. Word ngram language models and the lattice of candidates generated by the error model are used to decide the best alternative. This approach obtains the highest performance on the detection task of the ICDAR2017 competition [25], and improves the quality of OCRed text in both of two competitions in ICDAR2017 [25] and ICDAR2019 [135].

Instead of using the conventional error model, Sariev et al. [139] and Niklas [117] deploy word ngram language model with other techniques of candidate generation. Given an erroneous token, Sariev et al. [139] generate correction candidates by REBELS [53], the most appropriate one is then selected relying on word bigram/trigrams frequencies. Niklas [117] utilizes the anagram hash algorithm and a new OCR adaptive method to search for the best matching proposals for erroneous OCR tokens. The proposed method first classifies characters into equivalence classes based on their similar shapes. Characters of the same class will share the same OCR-key. The key of the given

[9] https://www.statmt.org/europarl/, accessed July 30, 2020.

word is used to retrieve all words that has the same key in the dictionary. In addition, word bigrams are applied to score suggested words.

As an alternative for separating error and language model, Kolak et al. [81] develop an end-to-end generative probabilistic model specifically designed for post-processing OCR output. The process starts by producing the true word sequence W, then present W as character sequence C. Next, the sequence C is segmented into p subsequences $\langle C^i, \ldots, C^p \rangle$ with $C^i = (C_{a_{i-1}}, \ldots, C_{a_i}]$ by a segmentation vector $a = \langle a_1, \ldots, a_{p-1} \rangle$ where $a_i < a_{i+1}, a_0 = 0, a_p = n$, and n as the length of C. Finally, each subsequence C^i is transformed into an OCRed subsequence O^i with b as a segmentation vector of O. Error correction is performed by finding the most probable source sequence $\hat{W}$ for an observation $(\hat{O}, \hat{b})$. Although this generative model is general, the authors only consider single-character edits in the implementation.

$$\hat{W} = \underset{W}{\operatorname{argmax}}\{\max_{a,C}[P(\hat{O}, \hat{b}|a, C, W) \times P(a|C, W) \times P(C|W) \times P(W)]\}. \tag{4}$$

Some methods of this type rely on Google Web 1T ngram corpus [18] for fixing errors, i.e., [9, 21, 147]. Bassil et al. [9] first identify *non-word* errors using word unigram frequency. Second, candidates are generated by word unigram and a character bigram frequency. Last, they choose the best alternative for each detected error relying on word 5-gram frequency. Soni et al. [147] concentrate on handling segmentation errors via Google 1T Web ngrams. They determine whether a token should be segmented based on the probability of word unigram, and bigram. Next, higher-order ngram probability is applied for deciding which tokens are likely to appear in context. One recent work of Cacho [21] identifies OCR errors relying on OCRSpell [154], and suggests candidates for each erroneous token based on word trigrams. The candidate with the highest frequency is used to substitute the error.

(1b) Neural network-based language models. Besides statistical language models, some approaches exploit neural network-based language models to post-correct OCRed text. Neural network language models learn to associate each lexical word with a continuous-valued vector of features (i.e., word embedding). Typically, they are constructed and trained as probabilistic classifiers to predict the probability distribution over the next word, given its context [12]. Along with word-level models, neural network language models can work at character level as well.

A character-level bidirectional LSTM language model is developed and applied to post-process digitised French clinical texts by D'hondt et al. [39, 40]. Given clean texts (i.e., digital collections containing no errors), they automatically create the training material by randomly applying edit operations (i.e., deletion, insertion, substitution). According to their results, the best performing systems outperform the baseline TICCL [132]. The authors also note that their models are not good at detecting errors related to word boundary.

In contrast to these above approaches that only work at character level, Magallon et al. [97] combine the characteristics of models at character and word levels. Features of character-level model are used as the input of the word-level model. It is also one of the participants of the ICDAR2017 competition [25]. Their performance is comparable to other approaches in the detection task.

(2) Feature-based machine learning models. Machine learning approaches learn from different features, so that more robust candidate selection is possible. These approaches explore multiple sources to generate candidates, extract features and rank them using a statistical model.

Kissos and Dershowitz [78] check each token in the input document against a lexicon to detect errors. For each error, an error model allowing multiple-character edits is used to generate its possible candidates. They then rank candidates by a regression model on four features: confusion weight obtained from their training data, word frequency, and forward/backward word bigram frequencies. These features and two additional ones (OCR confidence, term frequency in an OCRed

document) are used in the regression model to determine whether the OCRed token should be replaced by its highest ranked candidate. However, OCR confidence is not always available in OCRed text collections, therefore this model cannot fully be implemented. In addition, this approach does not consider an important feature employed in error correction [66], which is the similarity between an error and its candidate. The similarity between two strings (X, Y) can be measured based on their LV distance, or on their **longest common subsequence (LCS)**, which is a subsequence of both X and Y, and any sequence longer than Z is not a subsequence of X or Y [5], or on other ways.

Arguing that the above method uses solely ngram frequencies, Mei et al. [99] develop a similar approach with some supplemental features related to characteristics of OCR errors such as similarity metrics. Other features used in this method are word frequency, the existence in domain-dependent lexicon, word ngram frequency, and word skip-gram frequency. An OCRed token is identified as an error if its frequency is less than a threshold or the frequency of its context in the input document is not higher than another threshold. For each error, they suggest lexicon entries that differ from a given error within a limited number of character transformations as its correction alternatives. The authors introduce an updated version of this approach [98]. In the work, Google Web 1T ngram corpus is used to generate and score candidates. However, both of these works ignore an important feature: confusion weight, which is used in several successful post-processing approaches such as in References [82, 89, 122].

Khirbat [76] classifies a token as being incorrect or correct by using these following features: a presence of non alpha-numeric text in the token, a presence of the token in a dictionary, whether frequency of the token and its context in the input document is greater than a threshold, and whether its word bigram frequency is higher than another threshold. The method suggests lexicon entries whose LV distances to a given error are less than a threshold as its correction candidates. Each candidate is scored by a global optimization algorithm called simulated annealing [77] on a weighted sum of LV distance and LCS. The highest-rank candidate that presents in Google Web 1T ngram corpus is suggested to replace the error.

Inspired by these above models, Nguyen et al. [113] produce a list of alternatives for each OCR error by the error model of multiple-character edits. These candidates are then scored by a regression model on feature sets (i.e., confusion weight, context probability given by statistical/neural language model, and selected important features from two related works [78, 99].) The experimental results show that this multi-modular approach is comparable to the ones of the participants in the ICDAR2017 competition [25]. In another work, Nguyen et al. [115] mainly focus on detecting erroneous tokens from OCRed text. For each token in the input text, its possible replacements are generated from four sources (or four candidate generation sets): character error model, local word trigram context where they subsequently select likely candidates conditioned on its two neighbours on the left, or its left and right neighbours, or its two neighbours on the right. The main idea of this error detector is that an OCRed token needs to prove to be a valid word via binary classification with feature values computed from its candidate set. Two novel important features are suggested, including a frequency of an OCRed token in its candidate generation sets and a modified version of a peculiar index. The peculiar index represents a level of non-existent or rare n-grams in a token and it is computed based on frequencies of character bigrams/trigrams of a given token.

One work of Cacho [20] applies OCRSpell [154] for detecting errors and generating their replacements. Afterwards, SVM classification with five features (i.e., edit distance, confusion weight, unigram frequency, and backward/forward word bigram frequencies) is used to select the best alternative. Recently, Nguyen et al. [112] also select the most relevant correction candidates based on common features, i.e., string similarity, word frequency, word ngram frequency, and confusion weight. After detecting errors via lexicon-lookup, the authors generate and rank candidates by a

stochastic optimization algorithm with an objective function as a weighted combination of these above features.

(3) Sequence-to-sequence (Seq2Seq) models. Some approaches consider OCR post-processing as **machine translation (MT)** task, which transforms OCRed text into the corrected one in the same language. In this section, we group them with respect to traditional and neural Seq2Seq models.

(3a) Traditional Seq2Seq models. A source sentence (s) is translated into its corresponding target sentence (t) according to the probability distribution $p(t|s)$. This probability distribution is estimated by the probability $p(s|t)$ that s is the translation of t and the probability $p(t)$ of seeing t in the target language. In the context of post-OCR processing, s and t are in the same language, s as OCRed texts, t as corrected texts.

Afli et al. [1, 2] successfully train **statistical machine translation (SMT)** system to reduce OCR errors of historical French texts. They conclude that word-level SMT systems perform slightly better than character-level systems for OCR post-processing [1], and the word-level systems outperform language model techniques [2]. However, it should be noted that they have a large training set of over 60 million words while many datasets used for post-OCR processing are much smaller.

Schulz and Kuhn [143] present a complex architecture named Multi-Modular Domain-Tailored for OCR post-processing of historical texts. Their approach combines many modules from a word-level (e.g., original words, spell checker, compounder, word splitter, text-internal vocabulary) to a sentence-level (i.e., SMT) for candidate suggestion. Then, the decision module of Moses SMT decoder [80] is used to rank candidates.

Schnober et al. [141] conduct experiments to compare the performance of neural network Seq2Seq models against traditional ones on four NLP tasks including OCR post-processing. In their report, the neural network system underperforms a system built with Pruned CRF on a training set of 72,000 misrecognized words, but outperforms that system on a smaller training set of 10,000 errors.

(3b) Neural network Seq2Seq models (or NMT-based models). Apart from SMT, NMT directly transforms the source sentence into the target sentence using a single large neural network. Motivated by the great success of SMT on post-processing OCR output and the development of NMT, many recent approaches [6, 58, 106, 109, 140] apply them to fix OCR errors.

Multiple open sources of NMT are available with handy guidelines, therefore, several recent approaches make a use of them for denoising OCR output. Nastase et al. [109] utilize a character-level Seq2Seq model for correcting word boundary errors. The training and test data are from the electronic submissions of the ACL collection. The texts are split into smaller sequences of length less than 100. Word-level data are transformed into character-level data by deleting blank spaces between words in the input data or replacing them with special symbols "##" in the output data. Both resulting data are added blank space between characters (except for "#"). They train the model on the training data to learn word boundaries represented by special symbols "##" and use the trained model to fix word boundary errors of the test data.

Dong and Smith [41] exploit the character-level Seq2Seq model to post-correct OCRed text with some extensive attention mechanisms. In particular, they detect duplicated texts in OCR corpora and use them as multiple-input texts of the same output. The hidden states generated by encoding multiple-input texts are combined in different ways for generating better target sequences at decoding time.

Mokhtar et al. [106] employ NMT models at both word and character levels. Furthermore, they introduce a technique for enhancing the accuracy of the NMT outputs. Their experimental results on three historical datasets reveal that NMT models work better than their counterparts that use SMT. In addition, character-level models outdo word-level ones, since the former can deal with

unseen words. The performance is even better when they apply an additional post-processing step on the output of these NMT models.

Some participants of the two competitions on post-OCR text correction [25, 135], i.e., Char-SMT/NMT, CLAM, **Context-based Character Correction (CCC)**, UVA, implement character-level machine translation model to correct OCR errors. CLAM and UVA depend on NMT while Char-SMT/NMT [6] combines NMT and SMT.

Amrhein and Clematide [6] use character-level NMT models with diverse additional features (e.g., time span, text type, language, glyph embeddings, error-focused, etc.). This approach is applied to post-process OCRed text by researchers of National Library of the Netherlands[10] with their public post-correction tool named Ochre.[11] The authors [6] report that SMT systems outperform NMT systems in error correction, while NMT systems obtain higher performance in error detection. Their complex ensemble models achieve the best result in the correction task of the ICDAR2017 competition [25].

The CCC method is the winner of the ICDAR2019 competition [135]. It fine-tunes the pretrained language model **Bidirectional Encoder Representations from Transformers (BERT)** [38] with some convolutional layers and fully-connected layers to identify OCR errors. Their correction model is an attention Seq2Seq model with fine-tuning BERT. Inspired by the winners of the two competitions [25, 135], Nguyen et al. [116] propose a post-OCR processing approach developed from BERT and a character-level NMT with some extensions, including static embeddings used in BERT, character embeddings applied in NMT, and post-filter on length difference.

A recent work of Schaefer and Neudecker [140] exploits a bidirectional LSTM to predict each character in an input sequence as an incorrect or correct one. A sequence contains two words on the left and one word on the right of a given word. It is then identified as erroneous one if at least one of its characters is labelled as an error. The incorrect sequence is rectified by character-level NMT models that are trained to learn to translate OCRed texts into respective GT texts.

4 EVALUATION METRICS, DATASETS, LANGUAGE RESOURCES, AND TOOLKITS

4.1 Metrics

Regarding the error detection task, the goal is to determine whether an OCRed token was correctly recognized or not. In other words, it is same as a binary classification. Three popular metrics (i.e., Precision, Recall, their harmonic mean as the F-score) are used to evaluate the performance of error detection methods.

$$Precision = \frac{TP}{TP + FP}; \quad Recall = \frac{TP}{TP + FN}; \quad F\text{-}score = \frac{2 \times Precision \times Recall}{Precision + Recall}, \tag{5}$$

where **true positives (TP)** as the number of errors in the set that are detected by the method; **false positives (FP)** as the number of valid words in the set that are detected as errors; **false negatives (FN)** as the number of errors in the set that are not detected.

As to the correction task, there are some common evaluation measures, including the error rate (character error rate as CER, word error rate as WER), the accuracy (character accuracy as CAC, word accuracy as WAC), Precision, Recall, F-score, **bilingual evaluation understudy (BLEU)** [118].

The widely-used metric is the ER that quantifies the minimum number of character/word operations (i.e., insertions, deletions and substitutions) required to transform the ground truth text

[10] https://lab.kb.nl/about-us/blog/newspaper-ocr-quality-what-have-we-learned, accessed August 5, 2020.
[11] https://docs.google.com/document/d/1ui1wFNwIcnTn5gLvDL8JnM7epsod1uVAZNo31Zg_YuM/edit#heading=h.a7pnmiauas28, accessed August 5, 2020.

into the OCR output,

$$ER = \frac{I + S + D}{N}, \tag{6}$$

where N is the total number of characters/words in the ground truth and I, S, D as the minimal number of character/word insertions, substitutions, and deletions, respectively.

Accuracy (AC) is also a popular measure as the percentage of characters/words that are properly corrected. It is computed as below:

$$AC = 1 - ER. \tag{7}$$

Using Precision, Recall or correction rate, and F-score for evaluating performance of a post-processing approach is proposed by Reynaert [130]. These metrics are computed as corresponding Equation (5) in which TP, FP, **true negatives (TN)**, and FN are defined according to the four possible cases that can occur after post-processing OCRed text, as below:

(1) wrong → correct: a wrong character/word is corrected by post-processing (TP).
(2) correct → wrong: a correct character/word is changed to a wrong one by post-processing (FP).
(3) correct → correct: a correct character/word is still correct by post-processing (TN).
(4) wrong → wrong: a wrong character/word is still wrong by post-processing (FN).

BLEU [118] is a metric applied to assess the quality of the output texts in machine translation. In some post-processing approaches based on MT techniques, BLEU is also used to verify their performances by deciding how similar the translated text (the corrected OCRed text) is to the reference text (the GT text).

Besides these mentioned-above metrics, in the two competitions on post-OCR text correction [25, 135], the organisers score the participant approaches based on the relative improvement (*RelImp*) between the CER of the OCR output and the average corrected distance (*avgDist*).

$$RelImp = \frac{CER - avgDist}{CER} \times 100, \tag{8}$$

$$avgDist(corr, gt) = \frac{\sum_{i=1}^{n} \sum_{j=1}^{m} w_{ij} \times LVdistance(c_{ij}, gt_i)}{N},$$

where N is the number of characters in the GT, w_{ij} is the likelihood of a candidate c_{ij} to be a GT word gt_i, n is a number of words, and m is the number of candidates of an OCRed token corresponding to gt_i.

Some evaluation tools are developed by prior works and be freely accessible to the community: ocrevalUAtion [23][12] that is developed by the IMPACT Centre of Competence, ISRI OCR evaluation frontiers Toolkit [133][13] (denoted as ISRI toolkit), and the one of the competition on post-OCR text correction [25].[14]

4.2 Benchmarks

Different datasets are used to verify performance of post-processing approaches. We group them into two types: private and public. Whereas the public ones are free and accessible in terms of both OCRed texts (or equivalent materials like OCR models enabling to reproduce OCRed texts) and their corresponding GT, the private ones are not available (i.e., neither OCRed texts nor GT

[12] ocrevalUAtion: https://github.com/impactcentre/ocrevalUAtion, accessed July 23, 2020.
[13] https://code.google.com/archive/p/isri-ocr-evaluation-tools/, accessed August 6, 2020.
[14] https://sites.google.com/view/icdar2017-postcorrectionocr/evaluation, accessed July 23, 2020.

Table 1. A Summary of Open Accessible Datasets

	Dataset	Genre	Language	Size	Time scope	OCR engine	Link
Project corpora	Text+Berg [54]	Book	French, German, Italian, Romansh	87,000 pages with 35.75 million words	1864–2009	Abbyy FineReader 7, Omnipage 17	http://textberg.ch/site/en/corpora/
	GT4HistOCR [148]	Book	German, Latin	313,173 pairs of line images and GT texts	15th–19th century	Abbyy Finereader, OCRopus	https://zenodo.org/record/1344132#.Xv8sf ZMzZWM
	DBNL-OCR [36]	—	Dutch	—	1776–1878	—	https://lab.kb.nl/dataset/dbnl-ocr-data-set
	OCR-GT [166]	Newspaper	Dutch	2,000 pages	1700–1995	Abbyy FineReader 8.1, 9.0, 10.0	https://lab.kb.nl/dataset/historical-newspapers-ocr-ground-truth
	OCR17 [51]	Diverse genres	French	30,000 lines	17th century	Kraken, Calamari [164]	https://zenodo.org/record/3826894#.Xxn0u_ gzZWM
Competition datasets	TREC-5 [73]	Journal	English	55,600 documents	1994	—	https://trec.nist.gov/data/t5_confusion.html
	ALTA2017 [107]	Newspaper	English	8,000 documents	1806–2007	Abbyy FineReader	https://www.kaggle.com/dmollaaliod/correct-ocr-errors
	ICDAR2017 [25]	Newspaper, book	English, French	12 million characters	1654–2000	—	https://sites.google.com/view/icdar2017-postcorrectionocr/dataset
	ICDAR2019 [135]	Newspaper, book, receipt	Bulgarian, Czech, Dutch, English, Finish, French, German, Polish, Spanish, Slovak	22 million characters	—	—	https://sites.google.com/view/icdar2019-postcorrectionocr/dataset
Evaluation datasets	RETAS [168]	Book	English, French, German, Spanish	160 books	—	Abbyy FineReader	http://ciir.cs.umass.edu/downloads/ocr-evaluation/
	Frankfurter OCR-Korpus [44]	Book	Latin	5,213 words	1844–1855	—	https://www.texttechnologylab.org/applications/corpora/
	MiBio [98, 99]	Book	English	84,492 words	1907	Tesseract	https://github.com/jie-mei/MiBio-OCR-dataset
	Overproof-2 [47]	Newspaper	English	159 articles, 49 thousand words	1842–1954	Abbyy FineReader	http://overproof.projectcomputing.com/ datasets/
	Overproof-3 [47]	Newspaper	English	49 articles, 18 thousand words	1871–1921	Abbyy FineReader	http://overproof.projectcomputing.com/ datasets/
	noisy-MUC3&4 [68]	Newspaper	English	1,297 newspapers	—	Abbyy FineReader	https://ao.univ-lr.fr/index.php/s/BCKWtlg6HgTNGcv
	RDD&TCP [41]	Newspaper, book	English	1384 issues (RDD) and 934 books (TCP)	1860–1865, 1500–1800	Abbyy FineReader	http://www.ccs.neu.edu/home/dongrui/ ocr.html
	ACL collection [109]	Scientific articles	English	18,849 articles	1965–2012	—	https://www.cl.uni-heidelberg.de/english/research/downloads/resource_pages/ACL_corrected/ACL_corrected.shtml

"—" denotes no given information.

can be obtained). We only present the detail of 17 public datasets along with their downloadable links. A summary of these datasets is shown in Table 1.

These datasets contain diverse document genres (e.g., books, newspapers, receipts, journals, scientific articles). They cover a wide time range in various languages. Most of them are written in English, others consist of documents in languages such as Dutch, French, German, Spanish, Latin, and so on. As distinct OCR software can produce different OCRed text, thus, it is essential to know which OCR engine is used for generating each dataset. Additionally, publication time is an important information, since historical spellings often differ from standard ones. However, the information is sometimes missing from the description of the datasets. Short description of each dataset is presented as below:

(1) The first five corpora listed in Table 1 are the results of some digitisation projects or researches that mainly focus on dataset. Text+Berg Corpus and GT4HistOCR include books in multilingual languages. OCR-GT contains newspapers, whereas OCR17 has texts in diverse genres.
The Text+Berg Corpus [54] is the large multilingual digitised collection of alpine texts. The OCRed texts are the merging output of Abbyy FineReader 7 and Omnipage 17. As of March 2011, the corpus had 35.75 million words of yearbooks spanning the period from 1864 to 2009. These yearbooks relate to a central theme of mountains. A part of this corpus is digitised by the work [30].[15]

[15]https://files.ifi.uzh.ch/cl/OCR19thSAC/, accessed August 5, 2020.

The GT4HistOCR [148] contains 313,173 line pairs of images and GT. The texts originate from books in German and Latin written in Fraktur font printed between the 15th and 19th century. Some pretrained OCR models are also included in this corpus. GT4HistOCR is composed of five sub-corpora: DTA19,[16] Early Modern Latin,[17] Kallimachos,[18] Reference Corpus ENHG,[19] and RIDGES.[20]

The DBNL-OCR and OCR-GT are Dutch text. The DBNL-OCR includes 220 texts printed from 1776 to 1878. The OCR-GT is the OCR ground-truth of a dataset of historical newspapers, whose texts were published between 1700 and 1995. It has 2,000 pages processed by Abbyy FineReader version 8.1, 9.0, 10.0.

The last dataset of this group is OCR17 [51] that contains 30,000 French text lines published in the 17th century. In contrast to other datasets, OCR17 is processed by distinct OCR software, i.e., Kraken,[21] and Calamari [164].

(2) The next four datasets stem from four competitions: Text Retrieval Conference TREC-5 confusion track [73], the competitions on post-OCR text correction in ICDAR2017 [25] and 2019 [135], and the 2017 ALTA Shared Task on correcting OCR errors [107].

TREC-5 [73] confusion track used a set of 55,600 documents of the 1994 edition of the Federal Register. In this corpus, there are three versions of texts: the GT, the OCRed texts of 5% CER, and the texts of 20% CER.

The 2017 ALTA Shared Task dataset [107] has around 8,000 English documents from the Trove Newspaper collection of the National Library of Australia, containing documents processed by Abby FineReader and published within the period from 1806 to 2007.

The ICDAR2017 competition dataset [25] was built in the context of the AMELIOCR project.[22] It consists of 12 millions OCRed characters of English and French newspapers spanning from 1654 to 2000. The ICDAR2019 competition dataset [135] is larger (22 million characters) and contains newspapers, books, manuscripts, shopping receipts in 10 European languages: Bulgarian, Czech, Dutch, English, Finish, French, German, Polish, Spanish, and Slovak.

(3) The last corpora are evaluation datasets obtained from research papers, with three book datasets (i.e., **Recursive Text Alignment (RETAS)** [168], MiBio [98, 99], Frankfurter OCR-Korpus [44]), three newspaper datasets (i.e., Overproof-2 [47], Overproof-3 [47], noisy-MUC3&4 [68]), a scientific article corpus (ACL collection [109]) and a mixed dataset (RDD&TCP [41]).

Regarding the three book collections, the RETAS [168] consists of 160 digitised books in four languages (i.e., 100 English, 20 French, 20 German, and 20 Spanish) from the Internet Archive,[23] and the corresponding GT texts from the Project Gutenberg.[24] The Frankfurter OCR-Korpus [44] includes OCRed version and GT texts of 5,213 Latin words of the book *Patrologiae cursus completus: Series latina* (1844–1855). The MiBio [98, 99] is from the book *Birds of Great Britain and Ireland* (1907) with 84,492 words processed by Tesseract.

[16] http://www.deutschestextarchiv.de/, accessed July 21, 2020.
[17] http://www.cis.lmu.de/ocrworkshop/, accessed July 21, 2020.
[18] http://kallimachos.de, accessed July 21, 2020.
[19] https://www.linguistics.ruhr-uni-bochum.de/ref/, accessed July 21, 2020.
[20] https://korpling.org/ridges, accessed July 21, 2020.
[21] https://dev.clariah.nl/files/dh2019/boa/0673.html, accessed July 23, 2020.
[22] Led by the National Library of France (Department of preservation and conservation) and the L3i laboratory (Univ. of La Rochelle, France).
[23] https://www.archive.org, accessed July 7, 2020.
[24] https://www.gutenberg.org, accessed July 7, 2020.

Three of the newspaper corpora contain English texts that are recognised by Abbyy FineReader. The Overproof-2 and Overproof-3 are evaluation datasets of Evershed et al. [47]. Overproof-2 dataset includes 159 articles of the Sydney Morning Herald (1842–1954) with total 49 thousand words. Overproof-3 is a collection of 49 articles from the Library of Congress's Chronicling America newspaper archive. The noisy-MUC3&4 [68] includes OCRed noisy texts of 1,297 news reports originating from the corpus used in the 3rd and 4th **Message Understanding Conference (MUC)**.

The last two datasets are written in English. The RDD&TCP [41] includes 1,384 issues of the Richmond Daily Dispatch (1860–1865) and 934 books (1500–1800) from the Text Creation Partnership, coming from the Chronicling America collection and the Internet Archive. The ACL collection [109] is created from 18,849 scientific articles published from 1965 to 2012. Most of errors in this dataset are incorrect word segmentation.

Most of the public datasets contain unaligned OCRed texts along with GT ones, except for two competition datasets (i.e., ICDAR2017 and ICDAR2019). OCRed text needs to be aligned with its corresponding GT text for assessing performance of post-OCR approaches as well as building confusion matrix. Various algorithms have been developed for sequence alignments [86] and applied in verifying OCR systems.

Rice [133] aligns the OCR output with its corresponding GT text using Ukkonen's algorithm [157] that works efficiently for short sequences, however, it is too expensive for long ones especially when transcripts have no information on line breaks or page breaks. Feng and Manmatha [48], RETAS [168], and Chiron et al. [26] subdivide the problems and solve them using less computation as well as memory space. Particularly, they detect unique words common in both texts and regard them as anchor points for dividing long sequences into shorter ones. Each resulting segment is independently aligned by HMM-based model [48], or LCS [26, 168]. However, it is worth noting that dividing sequences into sub-sequences sometimes leads to different evaluation results, especially if OCR software inserts (or deletes) characters/words when outputting OCRed text. For instance, we have an OCRed sequence ($ocrSeq$='1 2 6 7 8 9') and its corresponding GT one ($gtSeq$='1 2 3 4 8 6 7 8 9'), word accuracy is computed as $wac_1 = 5/9$ after aligning the whole $ocrSeq$ and $gtSeq$. With maximum sequence length as 5, we split $ocrSeq$ into $ocrSeq_1$='1 2 6 7 8', $ocrSeq_2$='9' and $gtSeq$ into $gtSeq_1$='1 2 3 4 8', $gtSeq_2$='6 7 8 9', then word accuracy is computed as $wac_2 = 3/9$. Since '3 4 8' is deleted from $gtSeq$, wac_2 is lower than wac_1.

4.3 Language resources and toolkits

Along with freely accessible datasets and evaluation tools, several prior works publish their source code. All of them are absolutely valuable resources for further studies in this field. We present here a list of open sources: a free cloud service for OCR [16],[25] Hämäläinen and Hengchen [58],[26] Ochre,[27] OpenOCRCorrect [138],[28] PoCoTo [159],[29] Schnober et al. [141],[30] Silfverberg et al. [145],[31] Smith [41],[32] Soni et al. [147],[33] and Schaefer and Neudecker [140].[34]

[25] https://spraakbanken.gu.se/en/projects/ocr-cloud, accessed July 29, 2020.
[26] https://github.com/mikahama/natas, accessed July 16, 2020.
[27] https://github.com/KBNLresearch/ochre, accessed July 23, 2020.
[28] https://github.com/rohitsaluja22/OpenOCRCorrect, accessed July 23, 2020.
[29] https://github.com/thorstenv/PoCoTo, accessed July 30, 2020.
[30] https://github.com/UKPLab/coling2016-pcrf-seq2seq, accessed July 28, 2020.
[31] https://github.com/mpsilfve/ocrpp, accessed July 23, 2020.
[32] https://github.com/Doreenruirui/ACL2018_Multi_Input_OCR, accessed July 3, 2020.
[33] https://github.com/sandeepsoni/whitespace-normalizer, accessed July 3 2020.
[34] https://github.com/qurator-spk/sbb_ocr_postcorrection, accessed January 18, 2021.

Furthermore, some public and helpful language resources that are utilized in the post-OCR processing approaches are grouped below according to their languages:

(1) English: Accessible archives,[35] Brown corpus [84], British National Corpus,[36] Corpus of Historical American English,[37] Carnegie Mellon University Recipe Database [155],[38] CMU dictionary,[39] Chronicling America collection of historic U.S. newspapers,[40] a parallel corpus for statistical machine translation (Europarl-v9 corpus),[41] the Eighteenth Century Collections Online Text Creation Partnership,[42] Google Web 1T ngram [18], Google book ngrams [100], Oxford English Dictionary,[43] Reuters-21578,[44] Trove Newspaper collection,[45] and 1T word benchmark [24].[46]
(2) German: OCR-D groundtruth,[47] DTA—Das Deutsche Textarchiv (German Text Archive).[48]
(3) Finnish: historical newspapers corpus—Digi[49] and early modern corpus Kotus.[50]
(4) Multilingual corpora: Project Gutenberg,[51] historical Luxembourg newspapers,[52] Internet Archive.[53]
(5) Multilingual language models, word embeddings with public code: BERT [38], Fasttext embeddings [15], GloVe embeddings [121], and Word2Vec [103].

Besides language resources, some toolkits are available online for free access. They are categorised as follows:

(1) Spell checker: Aspell[54] and Hunspell.[55]
(2) Finite-State Tranducer: AT&T FSM,[56] Helsinki Finite-State Technology,[57] OpenFST.[58]
(3) Statistical Language Modeling: Carnegie Mellon Statistical Language Modeling [29],[59] KenLM Language Model [62],[60] Machine Learning for Language Toolkit,[61] SRI Language Modeling.[62]

[35] https://www.accessible-archives.com/, accessed July 16, 2020.
[36] https://www.english-corpora.org/bnc/, accessed July 3, 2020.
[37] https://www.ngrams.info/download_coha.asp, accessed July 3, 2020.
[38] https://www.cs.cmu.edu/~ark/CURD/, accessed July 3, 2020.
[39] http://www.speech.cs.cmu.edu/cgi-bin/cmudict, accessed July 23, 2020.
[40] https://chroniclingamerica.loc.gov/data/ocr/, accessed July 21, 2020.
[41] https://www.statmt.org/europarl/, accessed July 3, 2020.
[42] https://textcreationpartnership.org/tcp-texts/ecco-tcp-eighteenth-century-collections-online/, accessed July 21, 2020.
[43] https://www.oed.com/, accessed August 6, 2020.
[44] http://www.daviddlewis.com/resources/testcollections/reuters21578/, accessed July 21, 2020.
[45] https://trove.nla.gov.au/newspaper/, accessed July 29, 2020.
[46] https://code.google.com/archive/p/1-billion-word-language-modeling-benchmark/, accessed July 28, 2020.
[47] https://ocr-d.de/en/data, accessed July 3, 2020.
[48] http://www.deutschestextarchiv.de/download, accessed January 20, 2021.
[49] http://digi.kansalliskirjasto.fi/, accessed July 16, 2020.
[50] http://kaino.kotus.fi/korpus/vks/meta/vks_coll_rdf.xml, accessed July 16, 2020.
[51] http://www.gutenberg.org/, accessed July 3, 2020.
[52] https://data.bnl.lu/data/historical-newspapers/, accessed August 5, 2020.
[53] https://archive.org/details/texts, accessed July 3, 2020.
[54] http://aspell.net/, accessed July 21, 2020.
[55] https://github.com/hunspell/hunspell, accessed July 21, 2020.
[56] https://www3.cs.stonybrook.edu/ algorith/implement/fsm/distrib/, accessed July 3, 2020.
[57] https://hfst.github.io/, accessed July 3, 2020.
[58] http://www.openfst.org/twiki/bin/view/FST/WebHome, accessed July 3, 2020.
[59] http://www.speech.cs.cmu.edu/SLM/toolkit.html, accessed July 3, 2020.
[60] https://kheafield.com/code/kenlm, accessed July 3, 2020.
[61] http://mallet.cs.umass.edu/, accessed July 3, 2020.
[62] http://www.speech.sri.com/projects/srilm/, accessed July 21, 2020.

(4) Statistical Machine Translation: Moses SMT [80],[63] GIZA++,[64] ISI ReWrite Decoder.[65]
(5) Neural Machine Translation: Nematus [144],[66] OpenNMT [79].[67]
(6) Text Alignment: RETAS [168].[68]
(7) Natural Language Processing: Gensim [128],[69] NLTK,[70] Scikit-learn [119],[71] spaCy.[72]

5 DISCUSSION

After presenting various post-OCR processing approaches along with discussing freely accessible datasets, evaluation metrics, evaluation tools, public language resources and toolkits, we will next discuss in this section the current trends of post-OCR processing and suggest some extensions.

To get a clear view about the trends of post-OCR processing methods, we first illustrate in Figure 4(a) the numbers of approaches falling into each group based on their publication years. It can be seen that the total number of approaches gradually increased from 1997–1999 to 2015–2017 and then to 2018–2020. A few manual collaborative approaches have been developed to benefit from the public effort to correct OCR errors, the last one published in 2016. There is a noticeable change in the dominant position between the isolated-word and context-dependent approaches in some recent years. Isolated-word approaches with merging OCR outputs, domain-dependent dictionaries, error models, and so on, might have already reached the saturation point, whereas context-dependent ones are highly supported by existing toolkits as well as neural network techniques, and continue to thrive. The detailed statistics of isolated-word and context-dependent approaches are illustrated in Figure 4(b) and (c).

Post-OCR processing is one of the important steps of converting printed texts into electronic form, especially for historical documents on which performance of OCR software significantly decreases due to old spellings, obsolete fonts (e.g., Fraktur, Antiqua), complicated layouts and poor physical quality of used material. The proposed approaches however often work on different datasets and use different evaluation metrics, therefore, it is difficult to compare their performances.

Three recent competitions on post-OCR text correction organised in ICDAR2017, ICDAR2019, and ALTA2017 are starting points of solving this problem by assessing submitted approaches on the same dataset with the same metrics, even using the same published evaluation tool. For better views on the performance of methods on each dataset, we provide a summary table (Table 2) that groups methods with respect to either public datasets or ones used by more than one paper. Since the authors can split datasets in different ways or use various amounts of data for training and testing their methods, it is impossible to compare and suggest the best-performing one. Therefore, we can only list them along with their reported performances.

After studying several papers of post-OCR processing, we recommend some guidelines and potential directions in the two following sub-sections.

[63] http://www.statmt.org/moses/, accessed July 3, 2020.
[64] http://web.archive.org/web/20100129040616/http://fjoch.com/GIZA++.html, accessed July 21, 2020.
[65] https://www.isi.edu/natural-language/software/decoder/manual.html, accessed July 27, 2020.
[66] https://github.com/EdinburghNLP/nematus, accessed July 3, 2020.
[67] https://opennmt.net/, accessed July 3, 2020.
[68] http://ciir.cs.umass.edu/downloads/ocr-evaluation/, accessed July 3, 2020.
[69] https://radimrehurek.com/gensim/, accessed July 29 2020.
[70] https://www.nltk.org/, accessed July 21, 2020.
[71] https://scikit-learn.org/stable/, accessed August 6, 2020.
[72] https://spacy.io/, accessed July 21 2020.

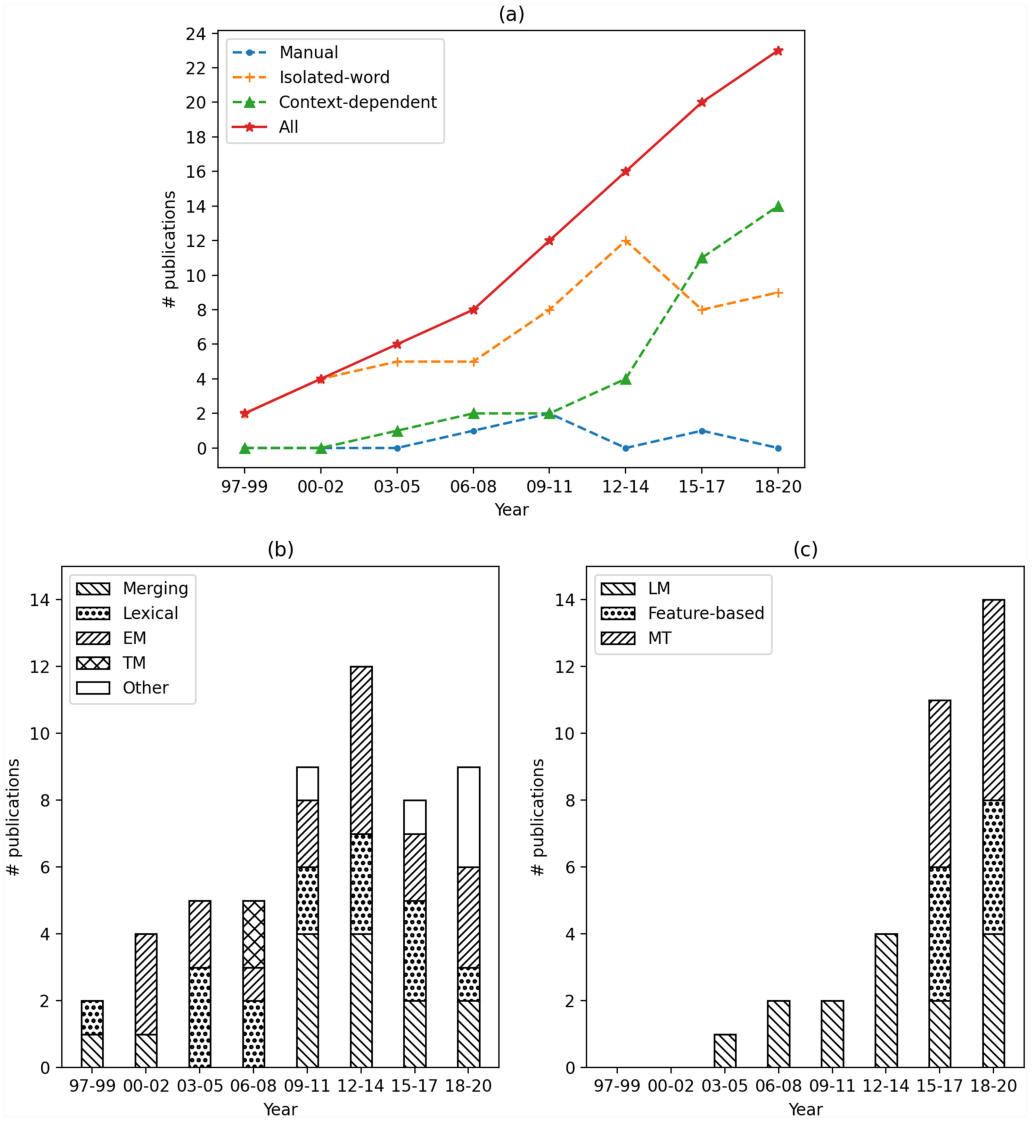


Fig. 4. Counts of post-OCR processing papers by their publication years (a). The papers are categorised into manual type and (semi-)automatic type that has two sub-types: isolated-word and context-dependent. The detailed statistics of papers of these two sub-types are shown in (b) and (c), respectively. We denote here "EM" as error models, "TM" as topic-based language models, "LM" as language models, "MT" as machine translation-based models.

5.1 Suggested guidelines

Among several techniques of (semi-)automatic post-OCR processing, we present below some suggestions to choose a suitable approach depending on dataset features.

(1) When the dataset is composed of pure text, approaches utilizing characteristics of the corpus itself and language resources (e.g., lexical methods, error-models, statistical language models, etc.) are the sole choices for cleaning noisy OCRed text. Most of the accessible datasets belong to this case.

Table 2. A Summary of Datasets and Methods That Used These Datasets for Evaluation

Dataset	Method	Year	Performance
ACL collection [109]	Nastase and Hitschler [109]	2018	P 95.5%, R 95%
ALTA2017 [107]	Khirbat [76]	2017	F 32.98%
Frankfurter OCR-Korpus [44]	Eger et al. [44]	2016	WAC 84.20%→90.16%
GT4HistOCR [148]	Englmeier et al. [45]	2019	WAC 65.03%→69.81%
ICDAR2017 dataset [25]	Schulz and Kuhn [143]	2017	CER (English: 28.4%→24.5%, French: 25.0%→21.5%), WER (English: 29.4%→22.1%, French: 13.3%→8.7%)
	Amrhein and Clematide [6]	2018	F (English: 67%, French: 50%), RelImp (English: 43%, French: 44%)
	Magallon et al. [97]	2018	F (English: 66%, French: 60%)
	WFST-PostOCR (competition team) [25]	2017	F (English: 73%, French: 69%), RelImp (English: 28%)
	CLAM (competition team) [25]	2017	F (English: 67%, French: 54%), RelImp (English: 29%, French: 5%)
	Nguyen et al. [113]	2018	RelImp (English:30%)
	Nguyen et al. [115]	2019	F (English: 79%)
	Nguyen et al. [116]	2020	F (English: 72%), RelImp (English: 36%)
	Nguyen et al. [112]	2020	F (English: 69%), RelImp (English: 33.7%)
ICDAR2019 dataset [135]	CCC (competition team)	2019	F (German: 95%), RelImp (German: 24%)
	CLAM (competition team)	2019	F (German: 93%), RelImp (Finnish: 44%)
	RAE1 (competition team)	2019	F (German: 90%), RelImp (French: 26%)
	Nguyen et al. [116]	2020	F (English: 68%), RelImp (English: 4%)
MiBio [98, 99]	Mei et al. [98, 99]	2016, 2017	CR 61.5%
noisy-MUC3&4 [68]	Jean-Caurant et al. [68]	2017	WER 46.51%→45.46%, CR (named entities) 37%
Overproof-2 [47]	Hammarström et al. [60]	2017	WAC 85.5%→86.5%
	Evershed and Fitch [47]	2014	WER 18.5%→6.3%
Overproof-3 [47]	Evershed and Fitch [47]	2014	WER 19.1%→6.4%
RDD&TCP [41]	Dong and Smith [41]	2018	CER 18.13%→7.73%, WER 41.78%→15.39%
Text+Berg [54]	Volk et al. [161]	2010	WAC 99.26%→99.35%
	Clematide et al. [30]	2016	WAC 99.71%, CAC 99.93%
	Schnober et al. [141]	2016	WAC 69.74%→74.67%
TREC-5 [73]	Mihov et al. [102]	2004	WAC 86.86%→97.03%
	Schulz et al. [142]	2007	R 95.95%
	Reffle et al. [126]	2009	WAC 98.86%→99.00%
	Gerdjikov et al. [53]	2013	CER 5%→4.3%
	Sariev et al. [139]	2014	WER 22.10%→3.27%, BLEU 58.44%→92.82%
	Hládek et al. [63]	2017	WER 52.31%→40.70%
	Cacho et al. [21]	2019	CR[73] 26%
	Cacho [20]	2020	F 71.49%, WAC 96.72%
Eisenhower Communiques [71] (English press release)	Lund and Ringger [93]	2009	WER 22.9%→14.7%
	Lund and Ringger [94]	2011	WER 19.88%→16.01%
	Lund and Walker [96]	2011	WER 18.24%→13.76%
	Lund et al. [95]	2014	WER 17.80%→16.23%
UWIII dataset [56] (English and Japanese journals)	Al Azawi and Breuel [3]	2014	ER 1.14%→ 0.68%
	Al Azawi et al. [8]	2014	CER 1.14%→0.80%
	Al Azawi et al. [4]	2015	CER 2.0%→0.40%
Digi corpus[74]	Chrons and Sundell [27]	2011	WAC 99%
	Kettunen et al. [75]	2014	WER 18.5%→6.6%
	Kettunen et al. [74]	2015	WAC 60.5%→70.1%, P 90%, R 49%, F 59%
DTA (German Text Archive)[75]	Schaefer and Neudecker [140]	2020	F 81%, CER 4.3%→3.6%
historical French texts [1]	Afli et al. [1]	2016	WER 4.9%→1.9%, BLEU 90.65%→96.74%
	Afli et al. [2]	2016	WAC 77.59%→80.48%, WER 22.41%→19.52%
Spanish name corpus [122]	Perez-Cortes et al. [122]	2000	WER 32.54%→1.74%, CER 6.36%→0.45%
	Llobet et al. [89]	2010	WAC 61.5%
	Perez-Cortes et al. [123]	2010	WAC 64.0%→78.5%
thematic English, German texts [150]	Strohmaier et al. [150]	2003	WAC (English: 98.95%→99.39%, German: 97.92%→98.69%)
	Strohmaier et al. [149]	2003	WAC 96.89%→97.43%
	Ringlstetter et al. [136]	2007	WAC (English: 91.71%→96.09%)
	Ringlstetter et al. [137]	2007	WAC (English: 89.26%→96.03%, German: 80.93%→88.48%)

We denote here "Year" as publication year, WAC as word accuracy, CAC as character accuracy, WER as word error rate, CER as character error rate, P as precision, R as recall, F as F1 score, CR as correction rate, RelImp as relative improvement, BLEU as bilingual evaluation understudy, $x \rightarrow y$ as x increases/reduces to y.

[73]This correction rate is computed as percentage of corrected errors over identified errors instead of all errors of the input.
[74]http://digi.kansalliskirjasto.fi/, accessed July 21, 2020.
[75]http://www.deutschestextarchiv.de/download, accessed January 20, 2021.

(2) When the corpus contains textual documents along with the character or word recognition confidences provided by OCR software, we can use this important information along with other sources in detecting suspicious tokens as well as correcting them.
(3) In the best case, if the original images are also available and re-OCRing is not too expensive, then we can take advantage from OCR results of different OCR engines as well as their recognition confidences. Besides the current OCR outputs merging approaches, the future post-processing systems may allow to select the best replacement from the candidate pools that are created from these OCR engines and generated from other sources (e.g., error model, word ngram frequency). Furthermore, context could be used to handle *real-word* errors.
(4) With a small dataset, an option is to use synthetic data for training post-processing models. Some techniques are proposed to generate artificial material such as randomly deleting, inserting, and substituting characters from a given word [39–41]; mimicking realistic errors from repetition texts: picking up an alternative from the list of frequent replaceable characters for a given character [57]; or using reversed error model with the input being GT word ngrams [52].

Focusing on models based on language resources, we highlight some of their noticeable features and helpful advice.

(1) The error model $P(s|w)$ plays a vital role in suggesting and ranking candidates. The central point of this model is the confusion matrix that could be estimated by aligning OCRed text and its corresponding GT. Since OCR errors can involve character segmentation, i.e., splitting characters (e.g., "the*m*" vs. "the*in*"), merging characters (e.g., "*li*nk" vs. "*b*nk"), it is important that the error model considers both single and multiple-character transformation.
(2) The language model $P(w)$ is the probability of the word w in the conventional corpus. The lexical approaches prove that domain-specific dictionaries or topic-dependent corpora can give more relevant scores. If the data collection contains old documents, then the methods need to consider historical language dictionaries, since historical spellings often do not conform to modern conventions.
(3) Instead of only focusing on the error model, the language model, we highly recommend combining these two models and other context information (e.g., word skip-ngrams, part-of-speech tagging, etc.) in feature-based machine learning models for obtaining higher performance.
(4) Neural network-based methods transform a given OCRed text into its corrected version by using implicit features. They often apply Seq2Seq with attention model or LSTM language model at character-level. Most of them do not use any character embeddings except the approach of the winner in the ICDAR2017 competition (Char SMT/NMT team) and the one of Nguyen et al. [116]. Moreover, these methods often work independently from external language resources such as Google Web 1T ngram, BERT language model, and so on. It is capable to enhance the performances of these methods if we utilize pre-trained embeddings or existing language resources. Some suggestions related to the network-based ones are reported in the next section.
(5) Considering post-OCR methods based on Seq2Seq models, NMT achieves good performance and it also dominates traditional Seq2Seq models in quantity, however, traditional Seq2Seq models still have their own advantages. In fact, NMT underperforms the Seq2Seq built with Pruned CRF conducted by Schnober et al. [141] and SMT outperforms NMT in error correction reported by Amrhein and Clematide [6]. We believe that improving the quality of OCRed text can benefit from a suitable combination of NMT and traditional Seq2Seq models.

5.2 Potential extensions

Along with the guidelines, we suggest some potential directions for developing post-processing approaches.

(1) Approaches based on machine learning require a lot of data to train a successful model, besides existing methods of generating synthetic data, we propose two more ways to create artificial materials. One opinion is to apply a machine translation model with the source text used as GT and target text used as OCRed text for obtaining more training data. Another suggestion is to use transfer learning: training a model on larger OCR corpora, then re-training it on a current data to learn its error distributions. Next, the re-trained model is used to generate artificial errors.

(2) Neural network-based approaches attract much attention from the community with promising performances. Their output can be improved if utilizing from external resources like pre-trained character or word embeddings, language models, and so on. We present some potential extensions on neural network as below:
- Although there are several popular word embeddings available (e.g., Fasttext [15] and GloVe [121]), public character embeddings, especially ones with old glyphs, are still missing. Character embeddings can be created to capture the similarity of character shapes or contexts, especially with old fonts, historical spellings.
- Neural network-based models could work at word-level if there are more data that may be generated by several above-mentioned techniques. A hybrid character-level and word-level model could possibly bring more benefits for post-OCR processing, although the initial evaluation [156] does not seem to support it.
- One choice is taking advantage of external resources in training models or post-processing their outputs. One simple but effective technique is to utilize these materials for filtering outputs of neural network-based approaches or to use them as the embedding input. More complex usage is possible to get higher performances.
- Another extension is, of course, to design specialized models for post-OCR processing. Besides neural network-based LM, Seq2Seq model with attention, SeqGAN and transformers might be some interesting options.

(3) A post-OCR processing approach can work effectively if both of its subtasks (i.e., error detection and correction) perform well. According to our observation, multiple approaches exploit simple techniques to detect errors, or even consider the list of errors as given. Nonetheless, it does not mean that error detection is not important, since one cannot fix errors without knowing their positions. Regarding the error correction task, there are few combinations between aforementioned models. We propose extensions based on each subtask as below:
- More attention should be paid to error detection methods. The future detector might not only identify errors but also classify them into groups, such as segmentation errors (e.g., "hometown" vs. "home town," "nice view" vs. "niceview"), recognition errors (e.g., "there" vs. "tberc"), and so on.
- Instead of relying on a single technique, we think that it is beneficial to combine candidates generated by several techniques: approximately searching, error models, language models, and machine translation models. For the candidate ranking, different methods could be tested: voting, feature-based ML, and so on.

(4) Moreover, it is necessary to have a deeper analysis on the performance of post-OCR processing approaches with OCRed texts generated from different OCR software on different features of OCR errors such as character segmentation, word segmentation, word length, and so on. More detailed information about OCR error types could be found in a

statistical research [114]. Additionally, interactive visualization and investigation tools that use large-scale language models could be useful for better understanding digitised documents, especially old ones. One example is DICT visualization approach (Document in the Context of its Time) that uses language models derived from the Google Books dataset that are temporally matched to the document publication date to visualize frequent/infrequent words as well as neologisms or obsolete terms that occur in the target document [67]. Such an analysis can become indispensable for researchers or developers to choose the relevant method for their data.

(5) From our observation, there are few post-OCR methods handling erroneous tokens involving word segmentation. This type of error accounts for around 18% of OCR errors [114] and happens when OCR software incorrectly recognizes spaces in documents. We think that future approaches could focus more on addressing this challenging error type.
(6) Regarding the evaluation metrics, equal weights are typically assigned to each word in documents. We think that it could be more meaningful if different weights are used depending on the importance of words especially when considering specific applications such as document retrieval.
(7) Among several datasets that are used to assess the performance of 91 papers, 17 datasets are freely accessible. They are valuable resources allowing researchers to compare performances and better understand the benefits and drawbacks of their methods. However, even with the same dataset, different ways of dividing training, development, or testing data could also lead to difficulties in effective comparison. Moreover, most of these used datasets lack some important information, such as publishing date of documents, OCR engines that were used to generate OCRed text, character or word recognition confidences, original images, and so on. We recommend some suggestions in dataset creation, namely, it is better to have a clear separation of data parts (i.e., training, testing, development data) for fair comparison; more detailed information (e.g., publication time, OCR engine, recognition confidences, original images) of each dataset could be useful for further studies.
(8) Since most of the current approaches are designated for post-processing English OCRed texts, it is essential for develop post-processing methods for other languages.

6 CONCLUSION

In this work we provide an extensive survey on approaches of post-correcting printed OCRed texts focusing mainly on English and other languages of Latin script. We first present the definition of the task and depict its typical pipeline. The importance of post-OCR processing is then motivated via several analyses of the impact of OCR errors toward downstream tasks. The post-processing approaches are grouped into manual and (semi-)automatic approaches, and then classified into isolated-word and context-dependent types depending on the extent of used information. For each type, its representative approaches are categorised into smaller groups depending on applied techniques. The characteristics of each main step, benefits, and drawbacks are then carefully summarized.

We report popular evaluation metrics and several freely accessible datasets in diversity of text types, languages, publication time. Some of the datasets come from the competition benchmarks that attract many participants with various methods. Future works in this field can benefit from these datasets and the existing experimental results. Some of the available datasets provide original images so that post-processing approaches can reOCR and take advantage from merging different OCR outputs as well as from the recognition confidences. Concerning the evaluation metrics, several measures have been applied to date, e.g., word/character error rate, word/character accuracy, Precision, Recall, F-score. Since these metrics weight all words equivalently, they may not be

always equally good to be applied for different downstream tasks that may need to put higher value to particular word types.

We next present discussions regarding the evolution of the methods of this field based on its progression since 1997 (after the last survey published about this field) with the noticeable trend in the development of context-dependent approaches over the recent years. Depending on the characteristics of the datasets, different methods are suggested to be applied. Neural network models have proven their abilities in post-correcting OCRed text with the best-performance on two multilingual competition datasets ICDAR2017 [25], ICDAR2019 [135] and a series of recent successful works. We believe that future research in this field will further look into this direction. Furthermore, since most of the approaches process English language data, a lot remains to be done for the post-processing approaches of other languages.

We hope that our work could be viewed as an important step toward more robust post-OCR processing techniques. Upcoming approaches on this task could not only focus on improving the effectiveness of correction but also concentrate on post-OCR processing in other languages, especially languages of other scripts (e.g., Arabic, Chinese), handwritten texts, and advanced neural network techniques.

A SUPPLEMENTARY MATERIAL

Supplementary online material contains a table that summarizes all the post-OCR processing approaches along with their performances.

REFERENCES

[1] Haithem Afli, Loïc Barrault, and Holger Schwenk. 2016. OCR error correction using statistical machine translation. *Int. J. Comput. Ling. Appl.* 7, 1 (2016), 175–191.

[2] Haithem Afli, Zhengwei Qiu, Andy Way, and Páraic Sheridan. 2016. Using SMT for OCR error correction of historical texts. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC'16)*. 962–966.

[3] Mayce Al Azawi and Thomas M. Breuel. 2014. Context-dependent confusions rules for building error model using weighted finite state transducers for OCR post-processing. In *Proceedings of the 2014 11th IAPR International Workshop on Document Analysis Systems*. IEEE, 116–120.

[4] Mayce Al Azawi, Marcus Liwicki, and Thomas M. Breuel. 2015. Combination of multiple aligned recognition outputs using WFST and LSTM. In *Proceedings of the 2015 13th International Conference on Document Analysis and Recognition (ICDAR'15)*. IEEE, 31–35.

[5] Lloyd Allison and Trevor I. Dix. 1986. A bit-string longest-common-subsequence algorithm. *Inform. Process. Lett.* 23, 5 (1986), 305–310.

[6] Chantal Amrhein and Simon Clematide. 2018. Supervised OCR error detection and correction using statistical and neural machine translation methods. *J. Lang. Technol. Comput. Ling.* 33, 1 (2018), 49–76.

[7] Richard C. Angell, George E. Freund, and Peter Willett. 1983. Automatic spelling correction using a trigram similarity measure. *Inf. Process. Manage.* 19, 4 (1983), 255–261.

[8] Mayce Ibrahim Ali Al Azawi, Adnan Ul-Hasan, Marcus Liwicki, and Thomas M. Breuel. 2014. Character-level alignment using WFST and LSTM for post-processing in multi-script recognition systems—A comparative study. In *Proceedings of the 11th International Conference on Image Analysis and Recognition ICIA(R'14)*, Lecture Notes in Computer Science, Vol. 8814. Springer, 379–386.

[9] Youssef Bassil and Mohammad Alwani. 2012. OCR context-sensitive error correction based on google web 1T 5-gram data set. *Am. J. Sci. Res.* 50 (2012).

[10] Youssef Bassil and Mohammad Alwani. 2012. OCR post-processing error correction algorithm using google's online spelling suggestion. *J. Emerg. Trends Comput. Inf. Sci.* 3, 1 (2012).

[11] Guilherme Torresan Bazzo, Gustavo Acauan Lorentz, Danny Suarez Vargas, and Viviane P Moreira. 2020. Assessing the impact of ocr errors in information retrieval. In *Proceedings of the European Conference on Information Retrieval*. Springer, 102–109.

[12] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *J. Mach. Learn. Res.* 3 (Feb. 2003), 1137–1155.

[13] Anurag Bhardwaj, Faisal Farooq, Huaigu Cao, and Venu Govindaraju. 2008. Topic based language models for OCR correction. In *Proceedings of the 2nd Workshop on Analytics for Noisy Unstructured Text Data (AND'08)* ACM International Conference Proceeding Series, Vol. 303. ACM, 107–112.

[14] Maximilian Bisani and Hermann Ney. 2008. Joint-sequence models for grapheme-to-phoneme conversion. *Speech Commun.* 50, 5 (2008), 434–451.
[15] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Trans. Assoc. Comput. Ling.* 5 (2017), 135–146.
[16] Lars Borin, Gerlof Bouma, and Dana Dannélls. 2016. A free cloud service for OCR/En fri molntjänst för OCR. Technical Report. Göteborg.
[17] Eugene Borovikov, Ilya Zavorin, and Mark Turner. 2004. A filter based post-OCR accuracy boost system. In *Proceedings of the 1st ACM Workshop on Hardcopy Document Processing*. 23–28.
[18] Thorsten Brants and Alex Franz. 2006. Web 1T 5-gram Version 1 LDC2006T13. In *Philadelphia: Linguistic Data Consortium*. Google Inc.
[19] Eric Brill and Robert C. Moore. 2000. An improved error model for noisy channel spelling correction. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 286–293.
[20] Jorge Ramón Fonseca Cacho and Kazem Taghva. 2020. OCR post processing using support vector machines. In *Intelligent Computing: Proceedings of the 2020 Computing Conference, Volume 2 (AI'20)*, Advances in Intelligent Systems and Computing, Vol. 1229. Springer, 694–713.
[21] Jorge Ramón Fonseca Cacho, Kazem Taghva, and Daniel Alvarez. 2019. Using the Google web 1T 5-gram corpus for OCR error correction. In *Proceedings of the 16th International Conference on Information Technology-New Generations (ITNG'19)*. Springer, 505–511.
[22] Ewerton Cappelatti, Regina De Oliveira Heidrich, Ricardo Oliveira, Cintia Monticelli, Ronaldo Rodrigues, Rodrigo Goulart, and Eduardo Velho. 2018. Post-correction of OCR errors using pyenchant spelling suggestions selected through a modified needleman–wunsch algorithm. In *Proceedings of the International Conference on Human-Computer Interaction*. Springer, 3–10.
[23] Rafael C. Carrasco. 2014. An open-source OCR evaluation tool. In *Proceedings of the 1st International Conference on Digital Access to Textual Cultural Heritage*. 179–184.
[24] Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2014. One billion word benchmark for measuring progress in statistical language modeling. In *Proceedings of the 15th Annual Conference of the International Speech Communication Association (INTERSPEECH'14)*. ISCA, 2635–2639.
[25] Guillaume Chiron, Antoine Doucet, Mickaël Coustaty, and Jean-Philippe Moreux. 2017. ICDAR2017 competition on post-OCR text correction. In *Proceedings of the 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR'17)*, Vol. 1. IEEE, 1423–1428.
[26] Guillaume Chiron, Antoine Doucet, Mickaël Coustaty, Muriel Visani, and Jean-Philippe Moreux. 2017. Impact of OCR errors on the use of digital libraries: Towards a better access to information. In *Proceedings of the 2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL'17)*. IEEE, 1–4.
[27] Otto Chrons and Sami Sundell. 2011. Digitalkoot: Making old archives accessible using crowdsourcing. In *Human Computation, Papers from the 2011 AAAI Workshop (AAAI Workshops)*, Vol. WS-11-11. AAAI.
[28] Kenneth W. Church and William A. Gale. 1991. Probability scoring for spelling correction. *Stat. Comput.* 1, 2 (1991), 93–103.
[29] Philip Clarkson and Ronald Rosenfeld. 1997. Statistical language modeling using the CMU-Cambridge toolkit. In *Proceedings of the 5th European Conference on Speech Communication and Technology*.
[30] Simon Clematide, Lenz Furrer, and Martin Volk. 2016. Crowdsourcing an OCR gold standard for a german and french heritage corpus. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC'16)*. 975–982.
[31] Ryan Cotterell, Nanyun Peng, and Jason Eisner. 2014. Stochastic contextual edit distance and probabilistic FSTs. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. 625–630.
[32] W. B. Croft, S. M. Harding, K. Taghva, and J. Borsack. 1994. An evaluation of information retrieval accuracy with simulated OCR output. In *Proceedings of the Symposium on Document Analysis and Information Retrieval*. 115–126.
[33] Fred J. Damerau. 1964. A technique for computer detection and correction of spelling errors. *Commun. ACM* 7, 3 (1964), 171–176.
[34] Dana Dannélls and Simon Persson. 2020. Supervised OCR post-correction of historical swedish texts: what role does the OCR system play? In *Proceedings of the Digital Humanities in the Nordic Countries 5th Conference* CEUR Workshop Proceedings, Vol. 2612. CEUR-WS.org, 24–37.
[35] Deepayan Das, Jerin Philip, Minesh Mathew, and C. V. Jawahar. 2019. A cost efficient approach to correct OCR errors in large document collections. In *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR'19)*. IEEE, 655–662.
[36] DBNL. 2019. DBNL OCR Data set.

[37] Andreas Dengel, Rainer Hoch, Frank Hönes, Thorsten Jäger, Michael Malburg, and Achim Weigel. 1997. Techniques for improving OCR results. In *Handbook of Character Recognition and Document Image Analysis*. World Scientific, 227–258.

[38] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT'19)*. Association for Computational Linguistics, 4171–4186.

[39] Eva D'hondt, Cyril Grouin, and Brigitte Grau. 2016. Low-resource OCR error detection and correction in french clinical texts. In *Proceedings of the 7th International Workshop on Health Text Mining and Information Analysis (Louhi@EMNLP'16)*. Association for Computational Linguistics, 61–68.

[40] Eva D'hondt, Cyril Grouin, and Brigitte Grau. 2017. Generating a training corpus for ocr post-correction using encoder-decoder model. In *Proceedings of the 8th International Joint Conference on Natural Language Processing (IJCNLP'17)*. Asian Federation of Natural Language Processing, 1006–1014.

[41] Rui Dong and David Smith. 2018. Multi-input attention for unsupervised OCR correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL'18)*. Association for Computational Linguistics, 2363–2372.

[42] Senka Drobac, Pekka Kauppinen, and Krister Lindén. 2017. OCR and post-correction of historical Finnish texts. In *Proceedings of the 21st Nordic Conference on Computational Linguistics*. 70–76.

[43] Senka Drobac and Krister Lindén. 2020. Optical character recognition with neural networks and post-correction with finite state methods. *Int. J. Doc. Anal. Recogn.* 23, 4 (2020), 279–295.

[44] Steffen Eger, Tim vor der Brück, and Alexander Mehler. 2016. A comparison of four character-level string-to-string translation models for (OCR) spelling error correction. *Prague Bull. Math. Ling.* 105 (2016), 77–100.

[45] Tobias Englmeier, Florian Fink, and Klaus U. Schulz. 2019. AI-PoCoTo: Combining automated and interactive ocr postcorrection. In *Proceedings of the 3rd International Conference on Digital Access to Textual Cultural Heritage*. 19–24.

[46] Paula Estrella and Pablo Paliza. 2014. OCR correction of documents generated during Argentina's national reorganization process. In *Proceedings of the 1st International Conference on Digital Access to Textual Cultural Heritage*. 119–123.

[47] John Evershed and Kent Fitch. 2014. Correcting noisy OCR: Context beats confusion. In *Proceedings of the 1st International Conference on Digital Access to Textual Cultural Heritage*. ACM, 45–51.

[48] Shaolei Feng and R. Manmatha. 2006. A hierarchical, HMM-based automatic evaluation of OCR accuracy for a digital library of books. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries (JCDL'06)*. ACM, 109–118.

[49] Florian Fink, Klaus U. Schulz, and Uwe Springmann. 2017. Profiling of OCR'ed historical texts revisited. In *Proceedings of the 2nd International Conference on Digital Access to Textual Cultural Heritage*. 61–66.

[50] Lenz Furrer and Martin Volk. 2011. Reducing OCR errors in gothic-script documents. In *Proceedings of the Workshop on Language Technologies for Digital Humanities and Cultural Heritage*. 97–103.

[51] Simon Gabay. 2020. OCR17: GT for 17th French prints.

[52] Michel Généreux, Egon W. Stemle, Verena Lyding, and Lionel Nicolas. 2014. Correcting OCR errors for german in fraktur font. In *Proceedings of the First Italian Conference on Computational Linguistics CLiC-It*. Pisa University Press, 186–190.

[53] Stefan Gerdjikov, Stoyan Mihov, and Vladislav Nenchev. 2013. Extraction of spelling variations from language structure for noisy text correction. In *Proceedings of the 12th International Conference on Document Analysis and Recognition*. IEEE, 324–328.

[54] Anne Göhring and Martin Volk. 2011. The Text+ Berg corpus: An alpine french-german parallel resource. In *Proceedings of Traitement Automatique des Langues Naturelles (TALN'11)*.

[55] Annette Gotscharek, Ulrich Reffle, Christoph Ringlstetter, and Klaus U. Schulz. 2009. On lexical resources for digitization of historical documents. In *Proceedings of the 9th ACM Symposium on Document Engineering*. ACM, 193–200.

[56] Isabelle Guyon, Robert M. Haralick, Jonathan J. Hull, and Ihsin Tsaiyun Phillips. 1997. Data sets for OCR and document image understanding research. In *Handbook of Character Recognition and Document Image Analysis*. World Scientific, 779–799.

[57] Kai Hakala, Aleksi Vesanto, Niko Miekka, Tapio Salakoski, and Filip Ginter. 2019. Leveraging text repetitions and denoising autoencoders in OCR post-correction. *CoRR* abs/1906.10907 (2019).

[58] Mika Hämäläinen and Simon Hengchen. 2019. From the paft to the fiiture: A fully automatic NMT and word embeddings method for OCR post-correction. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP'19)*. INCOMA Ltd., 431–436.

[59] Ahmed Hamdi, Axel Jean-Caurant, Nicolas Sidere, Mickaël Coustaty, and Antoine Doucet. 2019. An analysis of the performance of named entity recognition over ocred documents. In *Proceedings of the 2019 ACM/IEEE Joint Conference on Digital Libraries (JCDL'19)*. IEEE, 333–334.

[60] Harald Hammarström, Shafqat Mumtaz Virk, and Markus Forsberg. 2017. Poor Man's OCR post-correction: unsupervised recognition of variant spelling applied to a multilingual document collection. In *Proceedings of the 2nd International Conference on Digital Access to Textual Cultural Heritage*. 71–75.

[61] Andreas W. Hauser. 2007. *OCR-Postcorrection of Historical Texts*. Master's thesis. Ludwig-Maximilians-Universität München.

[62] Kenneth Heafield. 2011. KenLM: Faster and smaller language model queries. In *Proceedings of the 6th Workshop on Statistical Machine Translation*. 187–197.

[63] Daniel Hládek, Ján Staš, Stanislav Ondáš, Jozef Juhár, and Lászlo Kovács. 2017. Learning string distance with smoothing for OCR spelling correction. *Multimedia Tools Appl.* 76, 22 (2017), 24549–24567.

[64] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.* 9, 8 (1997), 1735–1780.

[65] Rose Holley. 2009. *Many Hands Make Light Work: Public Collaborative OCR Text Correction in Australian Historic Newspapers*. National Library of Australia.

[66] Aminul Islam and Diana Inkpen. 2009. Real-word spelling correction using Google Web IT 3-grams. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*. 1241–1249.

[67] Adam Jatowt, Ricardo Campos, Sourav S. Bhowmick, and Antoine Doucet. 2019. Document in context of its time (DICT): Providing temporal context to support analysis of past documents. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM'19)*. Association for Computing Machinery, New York, NY, 2869–2872.

[68] Axel Jean-Caurant, Nouredine Tamani, Vincent Courboulay, and Jean-Christophe Burie. 2017. Lexicographical-based order for post-OCR correction of named entities. In *Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition (ICDAR'17)*, Vol. 1. IEEE, 1192–1197.

[69] Sittichai Jiampojamarn, Colin Cherry, and Grzegorz Kondrak. 2010. Integrating joint n-gram features into a discriminative training framework. In *Human Language Technologies: Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 697–700.

[70] Hongyan Jing, Daniel Lopresti, and Chilin Shih. 2003. Summarization of noisy documents: A pilot study. In *Proceedings of the HLT-NAACL 03 on Text Summarization Workshop, Volume 5*. Association for Computational Linguistics, 25–32.

[71] D. R. Jordan. 1945. *Daily Battle Communiques*. Harold B. Lee Library.

[72] Philip Kahle, Sebastian Colutto, Günter Hackl, and Günter Mühlberger. 2017. Transkribus-A service platform for transcription, recognition and retrieval of historical documents. In *Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition (ICDAR'17)*, Vol. 4. IEEE, 19–24.

[73] Paul B. Kantor and Ellen M. Voorhees. 2000. The TREC-5 confusion track: Comparing retrieval methods for scanned text. *Inf. Retriev.* 2, 2-3 (2000), 165–176.

[74] Kimmo Kettunen. 2015. Keep, change or delete? Setting up a low resource ocr post-correction framework for a digitized old finnish newspaper collection. In *Proceedings of the Italian Research Conference on Digital Libraries*. Springer, 95–103.

[75] Kimmo Kettunen, Timo Honkela, Krister Lindén, Pekka Kauppinen, Tuula Pääkkönen, Jukka Kervinen, et al. 2014. Analyzing and improving the quality of a historical news collection using language technology and statistical machine learning methods. In *Proceedings of the IFLA World Library and Information Congress (IFLA'14)*.

[76] Gitansh Khirbat. 2017. OCR post-processing text correction using simulated annealing (OPTeCA). In *Proceedings of the Australasian Language Technology Association Workshop 2017*. 119–123.

[77] Scott Kirkpatrick, C. Daniel Gelatt, and Mario P. Vecchi. 1983. Optimization by simulated annealing. *Science* 220, 4598 (1983), 671–680.

[78] Ido Kissos and Nachum Dershowitz. 2016. OCR error correction using character correction and feature-based word classification. In *Proceedings of the 12th IAPR Workshop on Document Analysis Systems (DAS'16)*. IEEE, 198–203.

[79] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of the Association for Computational Linguistics on System Demonstrations (ACL'17)*. 67–72.

[80] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*. Association for Computational Linguistics, 177–180.

[81] Okan Kolak, William Byrne, and Philip Resnik. 2003. A generative probabilistic OCR model for NLP applications. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, Volume 1*. 55–62.

[82] Okan Kolak and Philip Resnik. 2002. OCR error correction using a noisy channel model. In *Proceedings of the 2nd International Conference on Human Language Technology Research*. Morgan Kaufmann Publishers Inc., 257–262.
[83] Okan Kolak and Philip Resnik. 2005. OCR post-processing for low density languages. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 867–874.
[84] Henry Kucera, Henry Kučera, and Winthrop Nelson Francis. 1967. *Computational Analysis of Present-Day American English*. Brown University Press.
[85] Vladimir I. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*.
[86] Heng Li and Nils Homer. 2010. A survey of sequence alignment algorithms for next-generation sequencing. *Brief. Bioinform.* 11, 5 (2010), 473–483.
[87] Xiaofan Lin. 2001. Reliable OCR solution for digital content re-mastering. In *Document Recognition and Retrieval IX*, Vol. 4670. International Society for Optics and Photonics, 223–231.
[88] Xiaofan Lin. 2003. Impact of imperfect OCR on part-of-speech tagging. In *Proceedings of the Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.* IEEE, 284–288.
[89] Rafael Llobet, Jose-Ramon Cerdan-Navarro, Juan-Carlos Perez-Cortes, and Joaquim Arlandis. 2010. OCR post-processing using weighted finite-state transducers. In *Proceedings of the 2010 International Conference on Pattern Recognition*. IEEE, 2021–2024.
[90] Daniel Lopresti. 2009. Optical character recognition errors and their effects on natural language processing. *Int. J. Doc. Anal. Recogn.* 12, 3 (2009), 141–151.
[91] Daniel Lopresti and Jiangying Zhou. 1997. Using consensus sequence voting to correct OCR errors. *Comput. Vis. Image Understand.* 67, 1 (1997), 39–47.
[92] William B. Lund, Douglas J. Kennard, and Eric K. Ringger. 2013. Combining multiple thresholding binarization values to improve OCR output. In *Document Recognition and Retrieval XX (SPIE Proceedings)*, Vol. 8658. SPIE, 86580R.
[93] William B. Lund and Eric K. Ringger. 2009. Improving optical character recognition through efficient multiple system alignment. In *Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries*. ACM, 231–240.
[94] William B. Lund and Eric K. Ringger. 2011. Error correction with in-domain training across multiple OCR system outputs. In *Proceedings of the 2011 International Conference on Document Analysis and Recognition*. IEEE, 658–662.
[95] William B. Lund, Eric K. Ringger, and Daniel David Walker. 2014. How well does multiple OCR error correction generalize? In *Document Recognition and Retrieval XXI*, Vol. 9021. SPIE, 76–88.
[96] William B. Lund, Daniel D. Walker, and Eric K. Ringger. 2011. Progressive alignment and discriminative error correction for multiple OCR engines. In *Proceedings of the 2011 International Conference on Document Analysis and Recognition*. IEEE, 764–768.
[97] Thibault Magallon, Frédéric Béchet, and Benoît Favre. 2018. Combining character level and word level RNNs for post-OCR error detection. In *Proceedings of the Actes de la Conférence TALN (CORIA-TALN-RJC'18), Volume 1*. ATALA, 233–240.
[98] Jie Mei, Aminul Islam, Abidalrahman Moh'd, Yajing Wu, and Evangelos E. Milios. 2017. Post-processing OCR text using web-scale corpora. In *Proceedings of the 2017 ACM Symposium on Document Engineering (DocEng 2017)*. ACM, 117–120.
[99] Jie Mei, Aminul Islam, Abidalrahman Moh'd, Yajing Wu, and Evangelos Milios. 2018. Statistical learning for OCR error correction. *Inf. Process. Manage.* 54, 6 (2018), 874–887.
[100] Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K Gray, Joseph P Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon Orwant, et al. 2011. Quantitative analysis of culture using millions of digitized books. *Science* 331, 6014 (2011), 176–182.
[101] Margot Mieskes and Stefan Schmunk. 2019. OCR quality and NLP preprocessing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*.
[102] Stoyan Mihov, Svetla Koeva, Christoph Ringlstetter, Klaus U. Schulz, and Christian Strohmaier. 2004. Precise and efficient text correction using levenshtein automata, dynamic Web dictionaries and optimized correction models. In *Proceedings of Workshop on International Proofing Tools and Language Technologies (2004)*.
[103] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of the 1st International Conference on Learning Representations (ICLR'13)*.
[104] David Miller, Sean Boisen, Richard Schwartz, Rebecca Stone, and Ralph Weischedel. 2000. Named entity extraction from noisy input: speech and OCR. In *Proceedings of the 6th Conference on Applied Natural Language Processing*. Association for Computational Linguistics, 316–324.
[105] Elke Mittendorf and Peter Schäuble. 2000. Information retrieval can cope with many errors. *Inf. Retriev.* 3, 3 (2000), 189–216.

[106] Kareem Mokhtar, Syed Saqib Bukhari, and Andreas Dengel. 2018. OCR error correction: State-of-the-Art vs an NMT-based approach. In *Proceedings of the 13th IAPR International Workshop on Document Analysis Systems (DAS'18)*. IEEE, 429–434.

[107] Diego Molla and Steve Cassidy. 2017. Overview of the 2017 ALTA shared task: Correcting ocr errors. In *Proceedings of the Australasian Language Technology Association Workshop 2017*. 115–118.

[108] Stephen Mutuvi, Antoine Doucet, Moses Odeo, and Adam Jatowt. 2018. Evaluating the impact of OCR errors on topic modeling. In *Proceedings of the International Conference on Asian Digital Libraries*. Springer, 3–14.

[109] Vivi Nastase and Julian Hitschler. 2018. Correction of OCR word segmentation errors in articles from the ACL collection through neural machine translation methods. In *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC'18)*. European Language Resources Association (ELRA).

[110] J. Ramon Navarro-Cerdan, Joaquim Arlandis, Rafael Llobet, and Juan-Carlos Perez-Cortes. 2015. Batch-adaptive rejection threshold estimation with application to OCR post-processing. *Expert Syst. Appl.* 42, 21 (2015), 8111–8122.

[111] J. Ramon Navarro-Cerdan, Joaquim Arlandis, Juan-Carlos Perez-Cortes, and Rafael Llobet. 2010. User-defined expected error rate in OCR postprocessing by means of automatic threshold estimation. In *Proceedings of the 12th International Conference on Frontiers in Handwriting Recognition*. IEEE, 405–409.

[112] Quoc-Dung Nguyen, Duc-Anh Le, Nguyet-Minh Phan, and Ivan Zelinka. 2021. OCR error correction using correction patterns and self-organizing migrating algorithm. *Pattern Anal. Appl.* 24, 2 (2021), 701–721.

[113] Thi-Tuyet-Hai Nguyen, Mickaël Coustaty, Antoine Doucet, Adam Jatowt, and Nhu-Van Nguyen. 2018. Adaptive edit-distance and regression approach for post-OCR text correction. In *Maturity and Innovation in Digital Libraries: Proceedings of the 20th International Conference on Asia-Pacific Digital Libraries (ICADL'18) (*Lecture Notes in Computer Science, Vol. 11279. Springer, 278–289.

[114] Thi-Tuyet-Hai Nguyen, Adam Jatowt, Mickaël Coustaty, Nhu-Van Nguyen, and Antoine Doucet. 2019. Deep statistical analysis of OCR errors for effective post-OCR processing. In *Proceedings of the 19th ACM/IEEE Joint Conference on Digital Libraries (JCDL'19)*. IEEE, 29–38.

[115] Thi-Tuyet-Hai Nguyen, Adam Jatowt, Mickaël Coustaty, Nhu-Van Nguyen, and Antoine Doucet. 2019. Post-OCR error detection by generating plausible candidates. In *Proceedings of the 2019 International Conference on Document Analysis and Recognition (ICDAR'19)*. IEEE, 876–881.

[116] Thi-Tuyet-Hai Nguyen, Adam Jatowt, Nhu-Van Nguyen, Mickaël Coustaty, and Antoine Doucet. 2020. Neural machine translation with BERT for Post-OCR error detection and correction. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020 (JCDL'20)*. ACM, 333–336.

[117] Kai Niklas. 2010. Unsupervised post-correction of OCR errors. *Master's thesis. Leibniz Universität Hannover* (2010).

[118] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. 311–318.

[119] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* 12 (2011), 2825–2830.

[120] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP'14)*. 1532–1543.

[121] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP'14)*. ACL, 1532–1543.

[122] Juan Carlos Perez-Cortes, Juan-Carlos Amengual, Joaquim Arlandis, and Rafael Llobet. 2000. Stochastic error-correcting parsing for OCR post-processing. In *Proceedings of the 15th International Conference on Pattern Recognition (ICPR'00)*, Vol. 4. IEEE, 405–408.

[123] Juan-Carlos Perez-Cortes, Rafael Llobet, J. Ramon Navarro-Cerdan, and Joaquim Arlandis. 2010. Using field interdependence to improve correction performance in a transducer-based ocr post-processing system. In *Proceedings of the 12th International Conference on Frontiers in Handwriting Recognition*. IEEE, 605–610.

[124] Alberto Poncelas, Mohammad Aboomar, Jan Buts, James Hadley, and Andy Way. 2020. A tool for facilitating ocr postediting in historical documents. In *Proceedings of the 1st Workshop on Language Technologies for Historical and Ancient Languages (LT4HALA'20)*. 47–51.

[125] Elvys Linhares Pontes, Ahmed Hamdi, Nicolas Sidere, and Antoine Doucet. 2019. Impact of OCR quality on named entity linking. In *Proceedings of the International Conference on Asian Digital Libraries*. Springer, 102–115.

[126] Ulrich Reffle, Annette Gotscharek, Christoph Ringlstetter, and Klaus U. Schulz. 2009. Successfully detecting and correcting false friends using channel profiles. *Int. J. Doc. Anal. Recogn.* 12, 3 (2009), 165–174.

[127] Ulrich Reffle and Christoph Ringlstetter. 2013. Unsupervised profiling of OCRed historical documents. *Pattern Recogn.* 46, 5 (2013), 1346–1357.

[128] Radim Řehůřek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50.

[129] Christian Reul, Uwe Springmann, Christoph Wick, and Frank Puppe. 2018. Improving OCR accuracy on early printed books by utilizing cross fold training and voting. In *Proceedings of the 13th IAPR International Workshop on Document Analysis Systems (DAS'18)*. IEEE, 423–428.

[130] Martin Reynaert. 2008. All, and only, the errors: More complete and consistent spelling and ocr-error correction evaluation. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC'08)*. European Language Resources Association.

[131] Martin Reynaert. 2008. Non-interactive OCR post-correction for giga-scale digitization projects. In *Proceedings of the International Conference on Intelligent Text Processing and Computational Linguistics*. Springer, 617–630.

[132] Martin W. C. Reynaert. 2011. Character confusion versus focus word-based correction of spelling and OCR variants in corpora. *Int. J. Doc. Anal. Recogn.* 14, 2 (2011), 173–187.

[133] Stephen Vincent Rice. 1996. *Measuring the Accuracy of Page-Reading Systems*. Ph.D. Dissertation. USA.

[134] Caitlin Richter, Matthew Wickes, Deniz Beser, and Mitch Marcus. 2018. Low-resource post processing of noisy OCR output for historical corpus digitisation. In *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC'18)*.

[135] Christophe Rigaud, Antoine Doucet, Mickaël Coustaty, and Jean-Philippe Moreux. 2019. ICDAR 2019 competition on post-OCR text correction. In *Proceedings of the 2019 International Conference on Document Analysis and Recognition (ICDAR'19)*. IEEE, 1588–1593.

[136] Christoph Ringlstetter, Max Hadersbeck, Klaus U. Schulz, and Stoyan Mihov. 2007. Text correction using domain dependent bigram models from web crawls. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'07) Workshop on Analytics for Noisy Unstructured Text Data*.

[137] Christoph Ringlstetter, Klaus U. Schulz, and Stoyan Mihov. 2007. Adaptive text correction with Web-crawled domain-dependent dictionaries. *ACM Trans. Speech Lang. Process.* 4, 4 (2007), 9.

[138] Rohit Saluja, Devaraj Adiga, Ganesh Ramakrishnan, Parag Chaudhuri, and Mark James Carman. 2017. A framework for document specific error detection and corrections in indic OCR. In *Proceedings of the 1st International Workshop on Open Services and Tools for Document Analysis and the 14th IAPR International Conference on Document Analysis and Recognition (OST@ICDAR'17)*. IEEE, 25–30.

[139] Andrey Sariev, Vladislav Nenchev, Stefan Gerdjikov, Petar Mitankin, Hristo Ganchev, Stoyan Mihov, and Tinko Tinchev. 2014. Flexible noisy text correction. In *Proceedings of the 2014 11th IAPR International Workshop on Document Analysis Systems*. IEEE, 31–35.

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

[141] Carsten Schnober, Steffen Eger, Erik-Lân Do Dinh, and Iryna Gurevych. 2016. Still not there? comparing traditional sequence-to-sequence models to encoder-decoder neural networks on monotone string translation tasks. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING'16)*. ACL, 1703–1714.

[142] Klaus U. Schulz, Stoyan Mihov, and Petar Mitankin. 2007. Fast selection of small and precise candidate sets from dictionaries for text correction tasks. In *Proceedings of the 9th International Conference on Document Analysis and Recognition (ICDAR'07)*, Vol. 1. IEEE, 471–475.

[143] Sarah Schulz and Jonas Kuhn. 2017. Multi-modular domain-tailored OCR post-correction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2716–2726.

[144] Rico Sennrich, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry, et al. 2017. Nematus: A toolkit for neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL'17)*. 65–68.

[145] Miikka Silfverberg, Pekka Kauppinen, Krister Linden, et al. 2016. Data-driven spelling correction using weighted finite-state methods. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics Proceedings of the SIGFSM Workshop on Statistical NLP and Weighted Automata*. ACL.

[146] David A. Smith and Ryan Cordell. 2018. *A Research Agenda for Historical and Multilingual Optical Character Recognition*. NUlab, Northeastern University (2018).

[147] Sandeep Soni, Lauren Klein, and Jacob Eisenstein. 2019. Correcting whitespace errors in digitized historical texts. In *Proceedings of the 3rd Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*. 98–103.

[148] Uwe Springmann, Christian Reul, Stefanie Dipper, and Johannes Baiter. 2018. Ground Truth for training OCR engines on historical documents in german fraktur and early modern latin. *J. Lang. Technol. Comput. Ling.* 33, 1 (2018), 97–114.

[149] Christian Strohmaier, Christoph Ringlstetter, Klaus U. Schulz, and Stoyan Mihov. 2003. A visual and interactive tool for optimizing lexical postcorrection of OCR results. In *Proceedings of the 2003 Conference on Computer Vision and Pattern Recognition Workshop*, Vol. 3. IEEE, 32–32.

[150] Christian M. Strohmaier, Christoph Ringlstetter, Klaus U. Schulz, and Stoyan Mihov. 2003. Lexical postcorrection of OCR-results: The web as a dynamic secondary dictionary?. In *Proceedings of the 7th International Conference on Document Analysis and Recognition*. Citeseer, 1133–1137.

[151] Kazem Taghva and Shivam Agarwal. 2014. Utilizing web data in identification and correction of OCR errors. In *Document Recognition and Retrieval XXI*, Vol. 9021. International Society for Optics and Photonics, 902109.

[152] Kazem Taghva, Julie Borsack, Allen Condit, and Srinivas Erva. 1994. The effects of noisy data on text retrieval. *J. Am. Soc. Inf. Sci.* 45, 1 (1994), 50–58.

[153] Kazem Taghva, Allen Condit, Julie Borsack, John Kilburg, Changshi Wu, and Jeff Gilbreth. 1998. Manicure document processing system. In *Document Recognition V*, Vol. 3305. International Society for Optics and Photonics, 179–184.

[154] Kazem Taghva and Eric Stofsky. 2001. OCRSpell: An interactive spelling correction system for OCR errors in text. *Int. J. Doc. Anal. Recogn.* 3, 3 (2001), 125–137.

[155] Dan Tasse and Noah A Smith. 2008. *SOUR CREAM: Toward Semantic Processing of Recipes*. Technical Report CMU-LTI-08-005 (2008).

[156] Konstantin Todorov and Giovanni Colavizza. 2020. Transfer learning for historical corpora: An assessment on post-OCR correction and named entity recognition. In *Proceedings of the Workshop on Computational Humanities Research (CHR'20)* , Vol. 2723. CEUR-WS.org, 310–339.

[157] Esko Ukkonen. 1995. On-line construction of suffix trees. *Algorithmica* 14, 3 (1995), 249–260.

[158] 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'20)*. SCITEPRESS, 484–496.

[159] Thorsten Vobl, Annette Gotscharek, Uli Reffle, Christoph Ringlstetter, and Klaus U. Schulz. 2014. PoCoTo-an open source system for efficient interactive postcorrection of OCRed historical texts. In *Proceedings of the 1st International Conference on Digital Access to Textual Cultural Heritage*. 57–61.

[160] Martin Volk, Lenz Furrer, and Rico Sennrich. 2011. Strategies for reducing and correcting OCR errors. In *Language Technology for Cultural Heritage*. Springer, 3–22.

[161] Martin Volk, Torsten Marek, and Rico Sennrich. 2010. Reducing OCR errors by combining two OCR systems. In *Proceedings of the ECAI 2010 Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaTeCH'10)*. 61–65.

[162] Luis Von Ahn, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum. 2008. recaptcha: Human-based character recognition via web security measures. *Science* 321, 5895 (2008), 1465–1468.

[163] David Wemhoener, Ismet Zeki Yalniz, and R. Manmatha. 2013. Creating an improved version using noisy OCR from multiple editions. In *Proceedings of the 2013 12th International Conference on Document Analysis and Recognition*. IEEE, 160–164.

[164] Christoph Wick, Christian Reul, and Frank Puppe. 2020. Calamari - A high-performance tensorflow-based deep learning package for optical character recognition. *Digit. Humanit. Q.* 14, 2 (2020).

[165] Michael L. Wick, Michael G. Ross, and Erik G. Learned-Miller. 2007. Context-sensitive error correction: Using topic models to improve OCR. In *Proceedings of the 9th International Conference on Document Analysis and Recognition (ICDAR'07)*. IEEE Computer Society, 1168–1172.

[166] L. Wilms, R. Nijssen, and T. Koster. 2020. Historical newspaper OCR ground-truth data set. KB Lab: The Hague.

[167] Shaobin Xu and David Smith. 2017. Retrieving and combining repeated passages to improve OCR. In *Proceedings of the 2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL'17)*. IEEE, 1–4.

[168] Ismet Zeki Yalniz and Raghavan Manmatha. 2011. A fast alignment scheme for automatic OCR evaluation of books. In *Proceedings of the 2011 International Conference on Document Analysis and Recognition*. IEEE, 754–758.

[169] Guowei Zu, Mayo Murata, Wataru Ohyama, Tetsushi Wakabayashi, and Fumitaka Kimura. 2004. The impact of OCR accuracy on automatic text classification. In *Advanced Workshop on Content Computing*. Springer, 403–409.

Received August 2020; revised January 2021; accepted February 2021