Vrije Universiteit Amsterdam

Universiteit van Amsterdam

Master Thesis

# Improving OCR Quality by Post-Correction

**Author:** Sara Salimzadeh (2614810, 12021520)

*1st supervisor:* Dr. Adam S.Z. Belloum
*daily supervisor:* Dr. Jaap Kamps
*2nd reader:* Dr. Giovanni Sileno

*A thesis submitted in fulfillment of the requirements for the joint UvA-VU Master of Science degree in Computer Science*

July 12, 2019

# Abstract

Currently, there is an intensive amount of research in the field of text digitization, especially with the growing interest in digital humanities. The main goal of such studies is converting scanned document images into searchable full text. To indexing the content of digitized historical archives, first, optical character recognition, known as OCR, is applied. However, the output of this process contains a significant amount of errors due to many reasons, such as the poor condition of historical documents, the variances of font sizes, lack of sufficient labeled data. This prompts the question if we can improve the accuracy of OCR-ed documents by performing post-processing. In the current thesis, we present a novel approach for automatic error detection and semi-automated error correction of OCR. Transfer learning and fine-tuning on top of seq2seq approach boost the model effectiveness and facilitate reusing OCR ground truth documents across languages. Thereby dramatically extending the value of expensive, human-annotated ground-truth training data.

Specifically, having designed a character level seq2seq model with a human assistant in the loop, we achieved to detect and correct more than 92% of errors in OCR documents. We also set up experiments to create a cross-lingual model with the help of fine-tuning and scored more than 50% for error detection. Furthermore, we studied the impact of OCR post correction on standard natural language processing end-to-end tasks. The analysis shows the excellence of OCR post-correction and demonstrates that even small improvements achievable by fully-automatic approaches have great value.



Keywords: OCR post-correction, natural language processing, historical documents, deep learning

# Acknowledgements

I would first like to thank my daily supervisor Prof. dr. Jaap Kamps of the Faculty of Humanities at the University of Amsterdam. He consistently allowed this project to be my work but steered me in the right direction.

I would also like to thank my supervisor Dr. Adam Belloum of the Institute of Informatics at the University of Amsterdam, who was involved in this research project. Without his passionate participation, the current thesis could not have been successfully conducted.

I would also like to acknowledge that the implementation of the thesis can be found in the following repository:

`https://github.com/sarasal/OCR_PostCorrection`.

# Contents

# List of Figures

## LIST OF FIGURES

# List of Tables

## LIST OF TABLES

# Chapter 1

# Introduction

There are lots of born-analog documents which contain valuable knowledge for society. For some decades, massive and expensive digitization of textual resources such as books, newspapers, historical archives has been underway, making these documents easily accessible [17]. Institutions are converting document images into machine-readable text via Optical Character Recognition, known as OCR, providing the opportunity for users exploring vast document corpora automatically, indexing for textual search and benefiting from machine translation. However, the quality of subsequent digital corpora can vary considerably depending on the quality of original paper, ink quality, differences in fonts, archaic vocabulary, text recognition techniques, etc. [10]

Unfortunately, the output of the OCR system, especially for historical documents is often faulty due to the poor physical deterioration of documents and limitation of OCR techniques. Commercial OCR systems (like Abbyy and OmniPage) and open-source OCR systems (like OCRopus and Tesseract) have been optimized for contemporary and end-user documents [6]. However, OCR engines for historical use cases differ in requirements from such traditional OCR systems mainly because of complex layouts.

Erroneous OCR-generated text and keywords corruption not only can make the text invisible to search tools and prevent users from retrieving relevant information but also lead to reading difficulties [13]. In general, the amount of noise in the corpora can affect the accuracy of subsequent negatively. Therefore, as the last activity in the OCR pipeline, post-processing tries to improve the performance by detecting and correcting errors [22].

Post-processing added to improve the quality of the output of the OCR systems. Different strategies have been applied for OCR post-processing classified into three groups: manual error correction, dictionary-based error correction, and context-based error correction. In the first type, humans are in charge of reviewing and editing the OCR-generated

text. However, this approach is costly in terms of time, and monetary invest and the result is still error-prone. The second method benefits from domain-specific lexicon to search for wrong words and replace them with correct ones automatically. Although this method can be easily implemented, it is not able to detect errors related to grammatical and semantic contexts and also out-of-vocabulary words. Another limitation is that conventional dictionary does not support names of regions, geographical locations, some technical keyword, and domain-specific terms [1]. Additionally, for specific low-resource domains, this method limits the usefulness. Finally, the third approach is proposed to cover the drawbacks of the first two methods and needing no external resources. The common strategies in this category are noisy channel model, statistical language model, statistical machine translation (SMT), neural machine translation (NMT) technique which translates OCR-generated text into the corrected text of the same language, and finally the ensemble of suggested methods.

Due to the lack of sufficient labeled data and the gold standard for training context-based approaches, instead of training different models from scratch, we can benefit from transfer learning [25] and fine-tuning. Having documents from various languages and genres, with the help of fine-tuning, we can tweak the parameters of an already trained network so that it adapts to the new data at hand.

The output of the OCR post-correction system could be fed into another pipeline such as standard NLP tasks for information retrieval, indexing, and further text analysis. As errors propagate from OCR post-correction to the later stages, to obtain acceptable efficiency for such tasks, we need to achieve relatively high performance in the post-correction system. In other words, It is crucial to find out how the improvement in the quality of OCR will be translated in boosting the performance of standard NLP tasks [14].

## 1.1 Main Problem and Research Questions

The main focus of the current research project is:

**Main Research Problem** *Can we improve Optical Character Recognition quality by post-correction?*

The born-analogue documents are digitized by making use of the OCR system. Nonetheless, the outcomes consist of many errors making following text analysis tasks error prone particularly for the historical texts. So, we feel the necessity of another component in between to fortify future text analysis tasks performance. With the help of the OCR post-correction module, we can produce digital texts closely enough to their analogue version

leading to getting more accurate results for searching, indexing, and information retrieval pipelines.

To solve our main problem, we split it into sub-questions as follows:

**RQ1** *What are the characteristics of OCR errors, and how can we classify them?*

We study the dataset in hand from different perspectives. To do so, we retrieve statistics regarding the number of errors, their length, frequency, and systematic errors. Having such information, we can gain better insight into the functionally of our solution.

**RQ2** *What is the performance of the seq2seq model on OCR post-correction?*

We are going to implement a seq2seq model to detect and correct OCR-ed documents on a relatively small corpus. Then, we will compare the result of such a model with different baselines to evaluate the performance, especially in the token level point of view.

**RQ3** *With limited training data, how can we take advantage of pre-trained models and transfer learning?*

The dataset contains different genres coming from various resources. Without the help of external resources such as large dictionaries, complex algorithms, and language statistics, we will study the usefulness of fine-tuning and transfer learning in our case to enhance the model performance.

**RQ4** *Can we even transfer OCR post-correction models across languages?*

It is trivial that using large corpora for model training leads to a more accurate model in terms of error detection and correction. However, we would like to observe how we can benefit from the dataset in hand to improve the model - taking this motivation into account, as a next step, we asses our model in the cross-lingual environment. We will carry out several experiments to check how language-independent our model is and how we can adapt it for more languages.

**RQ5** *How do improvement in OCR quality impact downstream NLP tasks, such POS-tagging and NER?*

After post-correction, the documents may go through another pipeline like question answering, inquiry in search engines, topic detection, etc. In such information retrieval applications, many NLP tasks are applied. To design desirable applications

with high-performance, we must reduce errors in early stages (the OCR system, the OCR post-correction system in our case). Therefore, at the end of the thesis, we determine the impact of OCR post-correction on standard end-to-end NLP tasks.

## 1.2 Related Work

There are many works and projects in the area of token level post- processing while the character level approach is quite novel. We will point out a subset of common methods in this section.

Based on Volk et al. [28], there are three approaches to improve the output of OCR system: Changing the input images, Optimizing the OCR system, Post-processing the output. In the current research, we focus on the last option. The main goal of OCR post-correction is detecting and correcting the errors related to both non-words and real-words in OCR-generated text. The common approaches for this task can be divided into the following categories:

### 1.2.1 Dictionary-based

Dictionary lookup compares OCR-output with the words in a lexicon. When there is a mismatch, one looks for alternatives within a small edit (Levenshtein) distance, under the assumption that OCR errors are often due to character insertions, deletions, and/or substitutions. Dictionaries can be used both in supervised and unsupervised methods [17]. The Text-Induced Corpus Clean-up (TICCL) system is an unsupervised corpus-based approach working on the historical spelling and correction for different languages [23]. To achieve high-efficiency, it requires high-quality lexicons. Koehn [18] benefited from the fixed vocabulary to train a neural model. The larger the vocabulary is, the less out-of-vocabulary words occur.

### 1.2.2 Context-based

**Language Model** A language model is just a probability distribution over sequences of words. This technique is founded on statistical language modeling and word n-grams. It aims at calculating the likelihood that a word sequence appears [3]. Atwell and Elliittm [5] have used a part-of-speech (POS) tagging method to detect the real-word errors in the text. WFST-PostOCR approach, one of the participants in the Post-OCR text correction competition 2017 [7], applied language model, and probabilistic character error model. They could achieve 28% improvement on the ICDAR 2017 error correction task.

**Noisy Channel Model** The aim of this approach is learning error models on the OCR-generated text combining error and language model. From a determination of the probability that a word will appear in the source document (the language model), and a determination of the probability of that word being corrupted into the observed OCR word (the error model) we can apply Bayes' rule and search for a candidate which maximizes the probability product[12]. Resnik [19] applied the first noisy channel model using edit distance from noisy to corrected text on the character level. Tong et al. [27] benefited from character n-gram and word bi-gram language model in addition to confusion probabilities to correct errors. Candidates are ranked by the conditional probabilities of matches given confusion matrix. The word bi-gram model and the Viterbi algorithm are used to determine the best scoring word sequence for the sentence. Evershed and Fitch [12] developed a system for automatic OCR post-correction based on the noisy channel approach. Their error model uses statistically weighted multiple character edits integrated with word 1-gram to 5-gram. At the heart of the error model, the confusion matrix illustrates the conditional probabilities of character edits using both single and pairs of characters for editing. Word correction candidates are the result of the heuristic search of weighted edit combinations and come from the language model or error model. Kissos and Dershowitz [17] also integrated a weighted confusion matrix with a shallow language model. The system is built out of two stages: 1) word expansion based on confusion matrix, 2) word selection using a regression model based on word features. The error correction methodology comprises three stages: 1) Correction Candidate Generation, 2) Feature Extraction in Word Level, 3) Word Classification. The experiments proved that the system could correct most frequent error types in Arabic dataset.

**Machine Translation** We can look into this problem from another point of view. OCR post-correction is a translation task converting OCR-generated text into corrected output in the same language. The recent line of research has tried to build models which correct errors automatically in this area. The problem can be viewed as s sequence to sequence task Sutskever et al. [26]. Among participants of ICDAR 2017 competition [7], Multi-Modular Domain- Tailored (MMDT) [24], Character Level Attention Model (CLAM), Character-based Statistical and Neural Machine Translation (Char-SMT/NMT) [4] are based on machine translation. CLAM and Char-SMT/NMT teams solutions rely on translation at character level. Char-SMT/NMT solution contains character-based SMT and NMT using Nematus framework combining with additional features such as factored NMT, Glyph embedding and, error-focused model. For character-based machine translation, is it required

to check the validity of a generated word before proposing as a candidate. Afli et. al. [2] trained a word-level SMT to correct OCR-ed historical French texts with 60 million tokens and got a better result than language model. They also indicated that machine translation at word level outperforms character level.

### 1.2.3 Ensemble

For a serious degree of corruption, the common approach is combining ensemble of multiple OCR engines. Multi-modular domain-tailored OCR post-correction (MMDT) system by Schulz and Kuhn [24] improves over single-handed methods. MMDT, another team competed in ICDAR 2017 [7], merged different modules from word level to sentence level (SMT) in order to propose candidates for correction and a decision module to rank them. The statistical machine translation includes both token and character level using Moses toolkit. Nguyen et al. [22] also obtain confusion matrix from the noisy channel model and context probability given by the language model. Their approach consists of three steps: candidate generating and weighting based on adaptive edit-distance, candidate scoring using language model, candidate ranking based on the regression model. The experimental results state that their approach is comparable to participants of ICDAR 2017. D'hont et al. [10] proposed a bidirectional recurrent neural network (encoder-decoder model) for learning a character-based language model without pre-annotated training dataset. Hokamp [15] also designed an ensemble of NMT models by incorporating features effective for word-level translation. Information such as Part-of-Speech (POS) tags, Name-Entity-Recognition (NER) labels can be included in the input of neural models to improve performance. Dong and Smith [11] proposed multi-input attention for unsupervised OCR correction in addition to the single-input attention-based sequence-to-sequence model. Multi-input attention combination is designed to correct multiple input sequences simultaneously and applied in the decoding step. In case of availability of multiple inputs, this approach outperforms supervised OCR correction in term of performance.

The Netherlands eScience Center situated in Amsterdam is the Dutch national center of excellence for the development and application of research software to advance academic research. Active projects carried out in eScience Center divided into five categories: sustainability and environment, life sciences and ehealth, physics and beyond, humanities and social sciences and, escience methodology. In the category of humanities and social sciences, Evaluation and Post-Correction of OCR of Digitised Historical Newspapers is one of the current projects of eScience Center. The simple version of this project (Ochre) exists.

Ochre is a toolbox for supervised OCR post-correction taking advantages of deep neural networks to correct noisy OCR-generated text. The pipeline of Ochre is as follows:

- Training character-based language model
- Applying post-correction
- Evaluation of results
- Performance Assessments
- Error analysis

Our work is directly motivated by the research question and general approaches of the Ochre toolbox.

## 1.3 Outline

In the next chapter, we study the characteristics of the dataset and address RQ1, followed by the Seq2seq Approach in the third chapter. Using the ICDAR 2017 dataset, we designed a character-level seq2seq model to identify the position of errors and correct them with the help of a human assistant concentrating on RQ2. Meanwhile, we participated in the ICDAR 2019 competition and submitted our result for English documents. We organized different experiments to discover the effect of using pre-trained model and fine-tuning. Chapter 4 includes our solution and analysis for RQ3. We go one step further to create the cross-lingual model out of our solution (RQ4), which is discussed in chapter 5. Afterward, we evaluated the impact of OCR post-correction on the standard natural language processing tasks and explained RQ5 in chapter 6. This report is ended up with the overall conclusion and future work.

# Chapter 2

# Data Analysis

In this chapter, we study our first research question: *What are the characteristics of OCR errors, and how can we classify them?* Given the broad overview of the dataset, we extract the information to get insight into the error numbers and error types. In addition to statistics from the character point of view, we also retrieve token-level statistics. Although our focus is character-level OCR post-correction, token-level statistics can help us evaluate the model performance from a different side.

## 2.1 Overview of Dataset

Through the current research, we focus on two datasets. Both experimental datasets come from OCR Post Correction Shared Task of International Conference on Document Analysis and Recognition (ICDAR); one is for the first edition in 2017, the other is obtained from the second edition in 2019. ICDAR competition consists of two tasks: 1) Detection of OCR Errors; 2) Correction of OCR Errors. For the first task, participants are asked to provide the position and length of suspected errors. In the second task, given the error in OCR-generated texts, participants are requested to suggest either only one correction or a ranked list of corrections for each error. The evaluation occurs in two phases: 1) fully automated: only one candidate is taken into account using a fully automatic approach; 2) semi-automated: with the help of human-assistant, the right candidate is picked from the list. (the higher-ranked the right correction, the better the system)

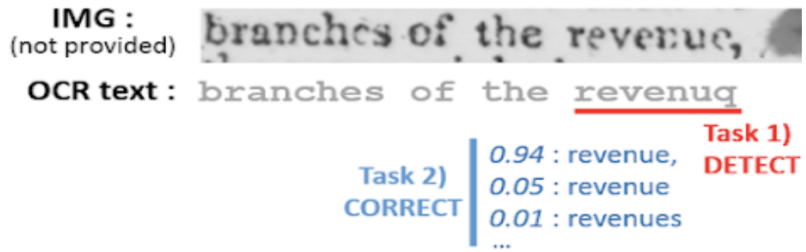



**Figure 2.1:** Tasks of ICDAR Competition: 1) Error Detection, 2) Error Correction

### 2.1.1 ICDAR 2017

ICDAR 2017 dataset is a subset of the corpus collected in the context of AmeliOCR project. This project is led by L3i laboratory (University of La Rochelle, France) and BnF (French National Library). The documents come from different collections (e.g., BnF, British Library) supported by various projects (e.g., Europeana Newspapers, IMPACT, Gutenberg, Perseus, Wikisource, and Bank of wisdom). This multi-lingual dataset contains French and English texts equally in the context of monographs and periodicals during the last four years. According to the Oxford Dictionary, we can define monographs and periodical as follows:

> Monograph: A detailed written study of a single specialized subject or an aspect of it.
>
> Periodical: A magazine or newspaper published at regular intervals.

It includes 12 million characters provided in three blocks: OCR-ed text, Gold Standard (GS) and, aligned OCR at the character level. Error rates depend on the nature of documents and could exceed 25%.

Training and Evaluation set are published separately by organizers of ICDAR. Training set composed of 666 files for English monographs and 58 files for English periodicals. In the evaluation set, there are 81 and 8 files for monographs and periodicals, respectively.

## 2.2 General OCR Error Analysis

OCR accuracy is negatively affected by the poor quality of scanned images. Depending on the language and image quality, the error distribution differs in OCR-generated texts.

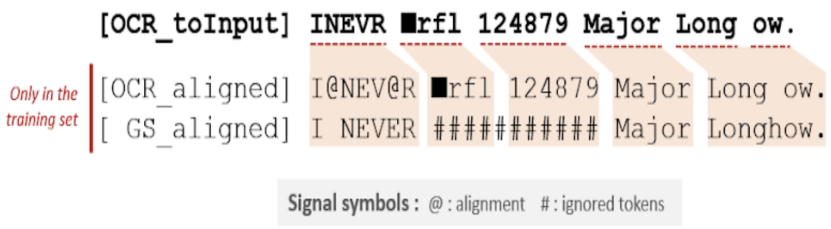



**Figure 2.2:** The Schema of ICDAR 2017 Dataset

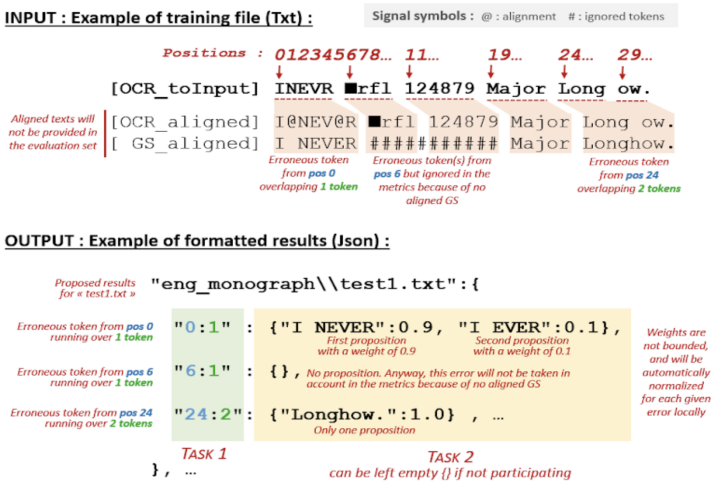



**Figure 2.3:** Tasks of ICDAR Competition: Example of Input and Output

Here, we provide standard categories of error types [1, 3, 8, 17], note that one error can belong to more than one group and error classes are not exclusive:

- **Word Detection**: Failing to detect text in the scanned image caused by the poor quality of the image or text mixed with graphics.
- **Word Segmentation**: Failing to bound the word correctly caused by wrong inter-word space detection.
- **Character Segmentation**: Failing to bound a single character within a segmented word due to the analog process which may disconnect connected components.
- **Character Recognition**: Failing to identify the correct character for a bounded character image.

## 2.3 ICDAR 2017 OCR Error Analysis

Based on the categorization above, we break down our analysis into token-level and character-level subgroups.

### 2.3.1 Statistics on Token Level

Token-level statistics consist of the number of tokens in the training and evaluation dataset, the pattern of erroneous tokens in terms of their length and the number of faulty characters, and also the number of consequent tokens forming n-grams. A **token** is a string of contiguous characters between two spaces. A token can present a single word or a group of words. We aligned OCR tokens to corresponding tokens in the ground truth and count how many pairs are not similar. Table 2.1 and 2.2 illustrate the number of tokens in training evaluation and dataset. Accordingly, the error rate in the training set is 7% in monographs and 15% in the periodicals. These rates are 11% and 15% in the evaluation data, respectively. Looking into the error rate, we can see that periodicals in the training set are a suitable sample of the evaluation set.

**Table 2.1:** Number of Tokens in English Training Set, ICDAR 2017

| Number of Tokens | Monograph | Periodical |
|---|---|---|
| Ground Truth | 625344 | 249541 |
| Original OCR | 650310 | 274479 |
| Erroneous | 43161 | 38496 |

**Table 2.2:** Number of Tokens in English Evaluation Set, ICDAR 2017

| Number of Tokens | Monograph | Periodical |
|---|---|---|
| Ground Truth | 136943 | 60458 |
| Original OCR | 138331 | 68579 |
| Erroneous | 15618 | 9788 |

Based on tables 2.3 and 2.4, even the occurrence of top-10 erroneous tokens in the training and evaluation set is infrequent. In the training set, the total frequency of systematic errors is less than 10% for monographs and 5% for periodicals; while in monographs, top-10 errors constitute 16% of errors and this measurement result is 3% in periodicals for the evaluation

set. Exploring such information, we can argue that the graphs of error distribution have long tales which make the error detection task a tough process.

As some errors contain the diacritics above, we can figure out the OCR system is not customized for English specifically - it seems it is tuned for French or the training data is mixed with French words. Furthermore, the system has low efficiency to distinguish between number and character as a considerable number of errors related to detection of '1' instead of 'I'. The amount of character misrecognition also leads to non-word error in which the recognized word doesn't correspond to any valid word in any dictionary.

**Table 2.3:** Top-10 Erroneous Tokens in English Training Set, ICDAR 2017

| Monograph | | | Periodical | | |
|---|---|---|---|---|---|
| OCR=>GS | Occurrence | Freq.(%) | OCR=>GS | Occurrence | Freq.(%) |
| 1=>I | 2403 | 5 | tbe=>the | 457 | 1 |
| the=>thé | 516 | 1 | tlie=>the | 277 | 0.7 |
| ail=>all | 151 | 0.3 | tho=>the | 181 | 0.4 |
| hut=>but | 130 | 0.3 | aund=>and | 177 | 0.4 |
| he=>be | 129 | 0.3 | ot=>of | 150 | 0.3 |
| aU=>all | 121 | 0.2 | he=>be | 120 | 0.3 |
| and=>And | 115 | 0.2 | tiie=>the | 115 | 0.3 |
| the=>THE | 104 | 0.2 | In=>in | 113 | 0.3 |
| what=>WHAT | 92 | 0.1 | iu=>in | 112 | 0.3 |
| corne=>come | 86 | 0.1 | ol=>of | 88 | 0.2 |

**Table 2.4:** Top-10 Erroneous Tokens in English Evaluation Set, ICDAR 2017

| Monograph | | | Periodical | | |
|---|---|---|---|---|---|
| OCR=>GS | Occurrence | Freq.(%) | OCR=>GS | Occurrence | Freq.(%) |
| 1=>1s | 1258 | 8 | tho=>the | 67 | 0.6 |
| the=>thé | 619 | 4 | tbe=>the | 48 | 0.4 |
| bas=>has | 136 | 0.8 | In=>in | 41 | 0.4 |
| ail=>all | 114 | 0.7 | aund=>and | 41 | 0.4 |
| hâve=>have | 105 | 0.6 | ot=>of | 27 | 0.2 |
| Thé=>The | 61 | 0.4 | iu=>in | 22 | 0.2 |
| aud=>and | 59 | 0.3 | ol=>of | 22 | 0.2 |
| waa=>was | 55 | 0.3 | 1=>I | 21 | 0.2 |
| bis=>his | 53 | 0.3 | Is=>1s | 20 | 0.2 |
| whioh=>which | 52 | 0.3 | he=>be | 20 | 0.2 |

The more incorrect characters a token has, the less probability it has to be corrected thoroughly. On the other hand, the longer the token is, it is more probable to guess the correct token since there is more context to do so. Therefore, having access to such information, we can optimize our solution to decline the number of token errors.

As the tokens are units which humans can recognize easily -instead of individual characters- it is informative to measure the effect of the model on reducing the number of faulty tokens. It is better to start with larger tokens and tokens having less incorrect characters. Figures 2.5 and 2.6 show that the rate of erroneous tokens consisting of 7 and more characters are higher than the rest both in the training and the evaluation set (steeper slope). Additionally, tokens comprised of 1 to 4 characters constitute the first half of the errors. We can also infer the training set is the appropriate representative of the evaluation set since both have the same trend.

**Figure 2.4:** Length of Erroneous Tokens in English Set, ICDAR 2017

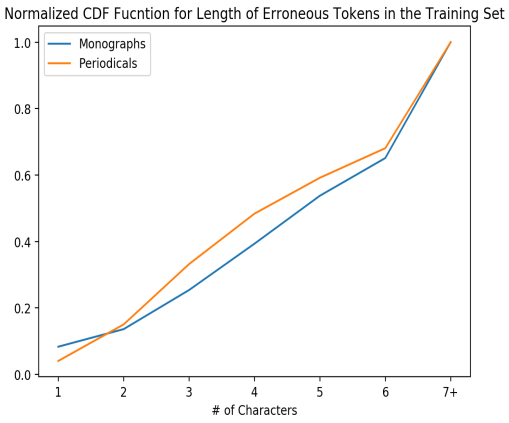



**Figure 2.5:** Training Set

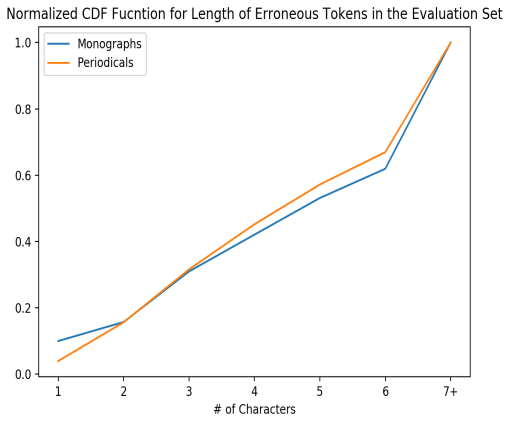



**Figure 2.6:** Evaluation Set

We also discover that more than half of the incorrect tokens have single character error, which makes the correction task easier, Figure 2.8 and 2.9. The majority of erroneous tokens have single character error in monographs while in periodicals, incorrect tokens distribute more homogeneously. Note that tokens with zero character error (x-axis) indicate the two proper tokens need a space insertion in between to be corrected. Alternatively stated, it measures the number of segmentation errors which happens when misrecognition of white-space occur on token level.

**Figure 2.7:** Number of Incorrect Characters within Erroneous Tokens in English Set, ICDAR 2017

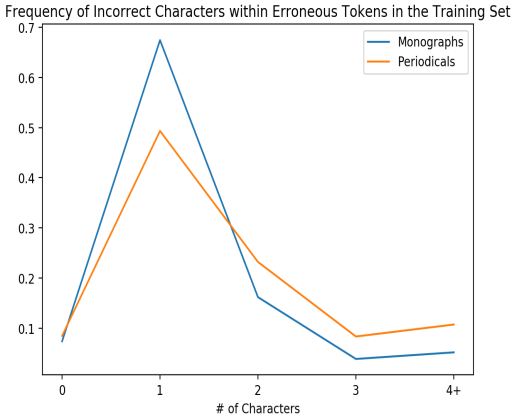



**Figure 2.8:** Training Set

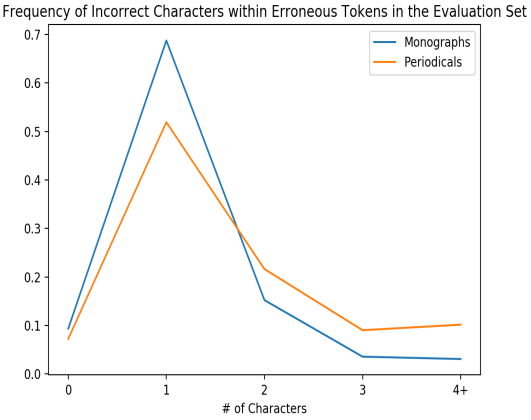



**Figure 2.9:** Evaluation Set

### 2.3.2 Statistics on Character Level

Since our focus on character level error detection and correction, we need relevant statistics in details. Tables 2.5 and 2.6 provide an overview of the training and evaluation dataset in terms of character numbers. The character error rate in the training set is 5% and 13% for monographs and periodicals retrospectively. These rates are 4% for monographs and 15% for periodicals in the evaluation set. These numbers illustrate that the rates of errors in the training and evaluation set are almost consistent.

**Table 2.5:** Number of Characters in English Training Set, ICDAR 2017

| Number of Characters | Monograph | Periodical |
|---|---|---|
| Ground Truth | 3592547 | 1574802 |
| Original OCR | 3592803 | 1575621 |
| Erroneous | 184998 | 206451 |

**Table 2.6:** Number of Characters in English Set, ICDAR 2017

| Number of Characters | Monograph | Periodical |
|---|---|---|
| Ground Truth | 781797 | 399700 |
| Original OCR | 781437 | 399033 |
| Erroneous | 31586 | 62057 |

We also derived the pattern of the most common character errors. Details are specified in the Tables 2.7 and 2.8. Additionally, Tables 2.9 and 2.10 show bigram errors. Accordingly, we can classify the errors as follows:

- **Character Format**: Variations in the font and similarity of character glyph can prevent accurate character recognition which leads to wrong word recognition such as [b,h], [0,O], [li,h], [c,e], [1,!]
- **Case Insensitivity**: It happens when lower- and upper-case characters are mixed up.
- **Punctuation Error**: Its happens when punctuation characters occur in the wrong places.
- **Segmentation Error**: It happens when misrecognition of white-space occurs on character and word level.
- **Character Insertion, Deletion, and Substitution (IDS)**: It occurs when one or more characters are substituted or deleted, or that a character is wrongly inserted in the middle of a word. It can lead to non-word error in which the recognized word does not correspond to any valid word in any dictionary.

The other issue is the low frequency (less than 1.5%) of systematic errors in proportion to the total number of error. In order to increase the rate of correction considerably, we may require a different method for each type of errors. We also look for more n-grams (n>2). However, the frequency of such cases is even less than bigrams. So, it doesn't lead to any added value.

**Table 2.7:** Top-10 Character Errors in English Training Set, ICDAR 2017

| Monograph | | | Periodical | | |
|---|---|---|---|---|---|
| OCR=>GS | Occurrence | Freq.(%) | OCR=>GS | Occurrence | Freq.(%) |
| d=>l | 618 | 0.3 | i=>h | 1012 | 0.4 |
| b=>h | 373 | 0.2 | c=>e | 392 | 0.2 |
| .=>, | 275 | 0.1 | e=>o | 343 | 0.1 |
| e=>E | 204 | 0.1 | I=>i | 277 | 0.1 |
| ,=>. | 200 | 0.1 | a=>n | 247 | 0.1 |
| 0=>O | 183 | < 0.1 | n=>m | 186 | < 0.1 |
| h=>H | 152 | < 0.1 | o=>O | 168 | < 0.1 |
| u=>n | 142 | < 0.1 | t=>T | 143 | < 0.1 |
| i=>I | 121 | < 0.1 | d=>D | 69 | < 0.1 |
| n=>N | 114 | < 0.1 | p=>P | 52 | < 0.1 |

**Table 2.8:** Top-10 Character Errors in English Evaluation Set, ICDAR 2017

| Monograph | | | Periodical | | |
|---|---|---|---|---|---|
| OCR=>GS | Occurrence | Freq.(%) | OCR=>GS | Occurrence | Freq.(%) |
| b=>h | 419 | 1 | o=>c | 168 | 0.2 |
| U=>l | 328 | 1 | a=>A | 94 | 0.1 |
| u=>n | 324 | 1 | s=>S | 80 | 0.1 |
| e=>c | 156 | 0.4 | l=>L | 56 | <0.1 |
| .=>, | 56 | 0.1 | n=>m | 55 | <0.1 |
| H=>i | 46 | 0.1 | e=>a | 52 | <0.1 |
| c=>e | 28 | <0.1 | t=>T | 45 | <0.1 |
| B=>R | 27 | <0.1 | r=>V | 32 | <0.1 |
| ,=>. | 25 | <0.1 | 6=>5 | 21 | <0.1 |
| n=>N | 14 | <0.1 | g=>G | 13 | <0.1 |

**Table 2.9:** Top-3 Erroneous Bigrams in English Training Set, ICDAR 2017

| Monograph | | | Periodical | | |
|---|---|---|---|---|---|
| OCR=>GS | Occurrence | Freq.(%) | OCR=>GS | Occurrence | Freq.(%) |
| d=>ll | 153 | <0.1 | he=>HE | 20 | < 0.1 |
| li=>h | 67 | <0.1 | es=>ES | 16 | < 0.1 |
| he=>HE | 31 | <0.1 | an=>AN | 14 | < 0.1 |

**Table 2.10:** Top-3 Erroneous Bigrams in English Evaluation Set, ICDAR 2017

| Monograph | | | Periodical | | |
|---|---|---|---|---|---|
| OCR=>GS | Occurrence | Freq.(%) | OCR=>GS | Occurrence | Freq.(%) |
| 1=>! | 103 | 0.3 | an=>AN | 17 | <0.1 |
| H=>li | 21 | < 0.1 | he=>HE | 16 | <0.1 |
| éâ=>ea | 19 | < 0.1 | rm=>RM | 14 | <0.1 |

From another point of view, we keep track of character insertion, deletion, and substitution required to correct original OCR input. Details are provided in Figure 2.10. If we skip unrecognizable characters, it is evident that substitution has the most significant number, in both training and evaluation set. The share of deletion and insertion is relatively identical. We see high occurrences of unrecognizable characters in periodicals. It is due to the nature of periodicals and the difficulty of the OCR system to recognize the text within the layout. Note that similar to the evaluation of ICDAR, tokens which are aligned with # symbol in the Gold Standard will be ignored in the metrics.

## 2.4 Conclusion

In this chapter, we studied our first research question: *What are the characteristics of OCR errors, and how can we classify them?* Our core dataset obtained from ICDAR 2017 composed of two genres, monographs and periodicals, in English and French. We focused on the English section in this chapter. We initially addressed the general approach for the OCR error analysis. By adjusting this framework into our use case, we broke down our study into token level and character level subsections for both genres. **Firstly**, we found that the error rate in the monographs and periodicals differ - more error in periodicals. More than half of the erroneous tokens consists of 1-4 characters. Furthermore, the number of tokens with a single character error is high relative to the rest. **Secondly**, moving on character level stats, we observed that the errors are distributed into Character Format, Case Insensitivity, Punctuation Error, Segmentation Error, and Character Insertion,

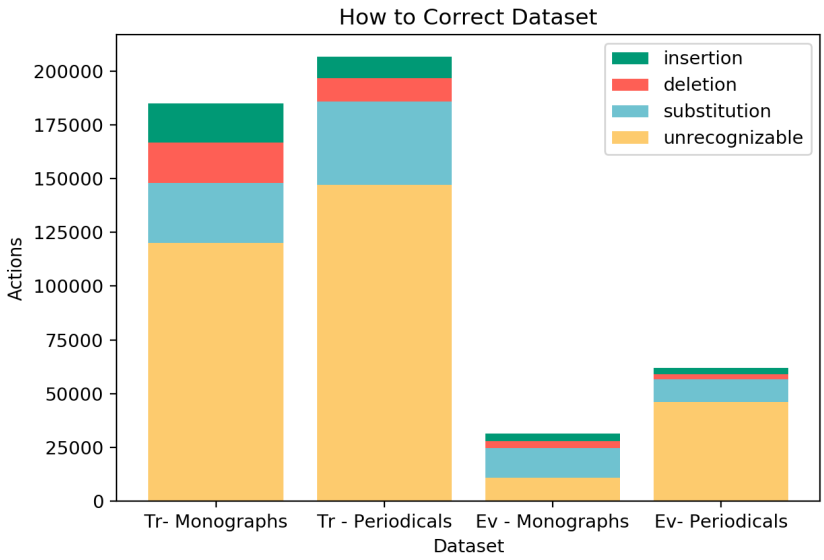



**Figure 2.10:** Actions Required to Correct English Documents

Deletion, and Substitution categories with the greatest portion for substitution. **Thirdly**, according to our first and second findings, we discovered that character level approaches would fit our case study the best.

# Chapter 3

# A Character-Level Seq2Seq Approach

In this chapter, we study the second research question: *What is the performance of the seq2seq model on OCR post-correction?* We are going to design a character level seq2seq model trained for the error detection task. According to the Pipeline 3.1, in the previous chapter, we examined the dataset. Now, we plan to explain the rest of the workflow: data pre-processing, model architecture, and data post-processing, respectively. In the end, we compare our output with the original OCR-ed text and analyze the results from different perspectives.

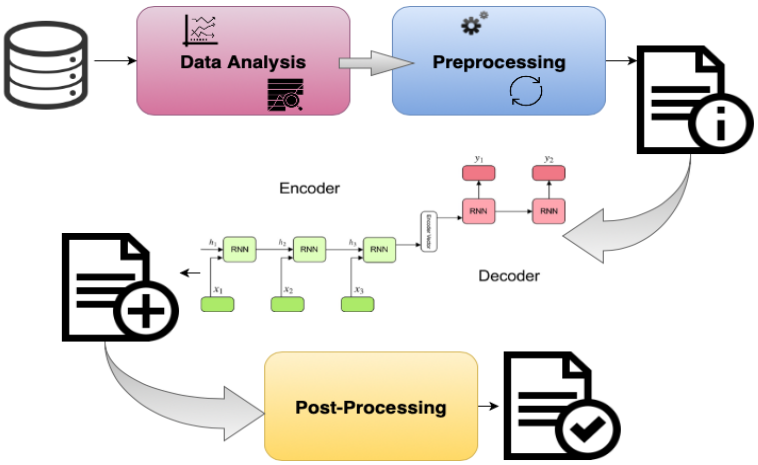



**Figure 3.1:** Our Pipeline for OCR Post-Correction

## 3.1 Motivation for Character level Neural Network

Deep learning architectures and algorithms have already made impressive advances in fields such as computer vision and pattern recognition. Following this trend, recent NLP research is now increasingly focusing on the use of new deep learning methods enabling multi-level automatic feature representation learning. For decades, shallow models in traditional machine learning (e.g., SVM and logistic regression) trained on very high-dimensional and sparse-feature NLP problems. In the last few years, neural networks produce superior results in many NLP tasks [29].

There are many systems in academia and industry focusing on token-level error correction. However, the size of the current dataset limits us to do so. Moreover, the common problem in languages with large dictionaries is the issue of out-of-vocabulary words. Character embedding deals with this issue naturally because words are the composition of individual letters. Making use of characters as units of the detection and correction task, we get rid of external dictionaries as well.

## 3.2 Pre-processing

In many cases, we observe that the datasets are not entirely clean. Since data comes from different sources, it has various characteristics which make preprocessing an essential task in the pipeline. In this step, we aim to transfer text into something that an algorithm can digest based on our problem.

Our model is based on character-level detection and correction. Each file of the dataset contains OCR input, aligned OCR, and aligned GS separated by a new line. We can see the symbol '@' within the aligned OCR and aligned GS. This character is used for aligning. We can convert aligned OCR to OCR input by removing such a symbol. Furthermore, there are high occurrences of the character '#' in aligned GS for unrecognizable characters of OCR input.

Before any cleaning operations, first, we load the files into the memory and align each pair of OCR and GS using two parameters: sequence length, step size. Sequence length specifies the number of characters within each sequence. Step size demonstrates the difference between two consequent sliding windows. In other words, each time, we shift equal to step size and separate characters for the next sequence. The following sample is provided as an example. According to Figure 3.2, every pair of OCR and GS consists of two fixed-length sequences of characters (except the last parts of the documents).

OCR_aligned:
My Helen is the fairest flower, That ever graced the sun or shade, Or deck'd with charms the lover's bovver, The desert wild, or sunny shade, Her bosoms fairer than the snow, Or April showers or May morns breath, Then moonlight rays or ruby's glow, Than weeping lily closed in death.

GS_aligned:
## Helen is the fairest flower, That ever graced the sun or shade, Or deck'd with charms the lover's bo@wer, The desert wild, or sunny shade, Her bosoms fairer than the snow, Or April showers or May morns breath, Then moonlight rays or ruby's glow, Than weeping lily closed in death.

step_size = 5
sequence_length = 15

pair 1: ['My Helen is the', '## Helen is the']
pair 2: ['len is the fair', 'len is the fair']
pair 3: ['s the fairest f', 's the fairest f']
.
.
.
pair n-1:['n death.', 'n death.']
pair n: ['th.', 'th.']

**Figure 3.2:** Sample of OCR and GS Alignment

Having all OCR and GS aligned, we discarded the symbol '@' and replaced '#' with space. Then, in order to force the neural model to learn the input character by character, we inserted a space between every two characters 3.3.

Before ['My Helen is the fairest flower', '## Helen is the fairest flower']
After ['M y H e l e n i s t h e f a i r e s t f l o w e r', '  H e l e n I s t h e f a i r e s t f l o w e r']

**Figure 3.3:** Space Insertion in OCR-GS Pairs

Characters can't be the input of the neural networks and we have to convert them into numeric values. So, we need to find a mapping between characters and integers. In a character embedding model, the vector for a sequence of characters is constructed from the character n-grams that compose it. Each character can be presented by its index within a vocabulary of letters. We make use of Keras tokenizer to fit on OCR sequences and create an internal vocabulary. Then, we substitute the characters by their indexes. The dictionary contains characters as entries and the indices are set based on their frequencies. Note that for sequences with characters less than the sequence length due to the lack of enough context for the last line of each document, we add padding. The tokenizer only encodes UTF-8 characters and skips the punctuations and spaces. It also normalizes all input characters to lowercase; therefore, we lose all of such information. Since white space is an identifier to separate tokens, we replaced it with a character from a different language (in our case, Farsi) to make the tokenizer assign an index for it in the vocabulary.

Similarly, the Keras models are not able to return characters as their output. We need to repeat the task above on GS sequences as well. The model functionality is predicting the probability of all characters in the vocabulary list for each character in the input sequences. Therefore, the format for output should be one-hot encoded. We pick the index with the greatest probability and insert its equivalent character. We iterate over all characters of a sequence to form output sequence. Using to_categorical function in Keras, we modify the GS to one-hot encoded format. Figure 3.4 summarizes the entire pre-processing of a OCR and GS pair.

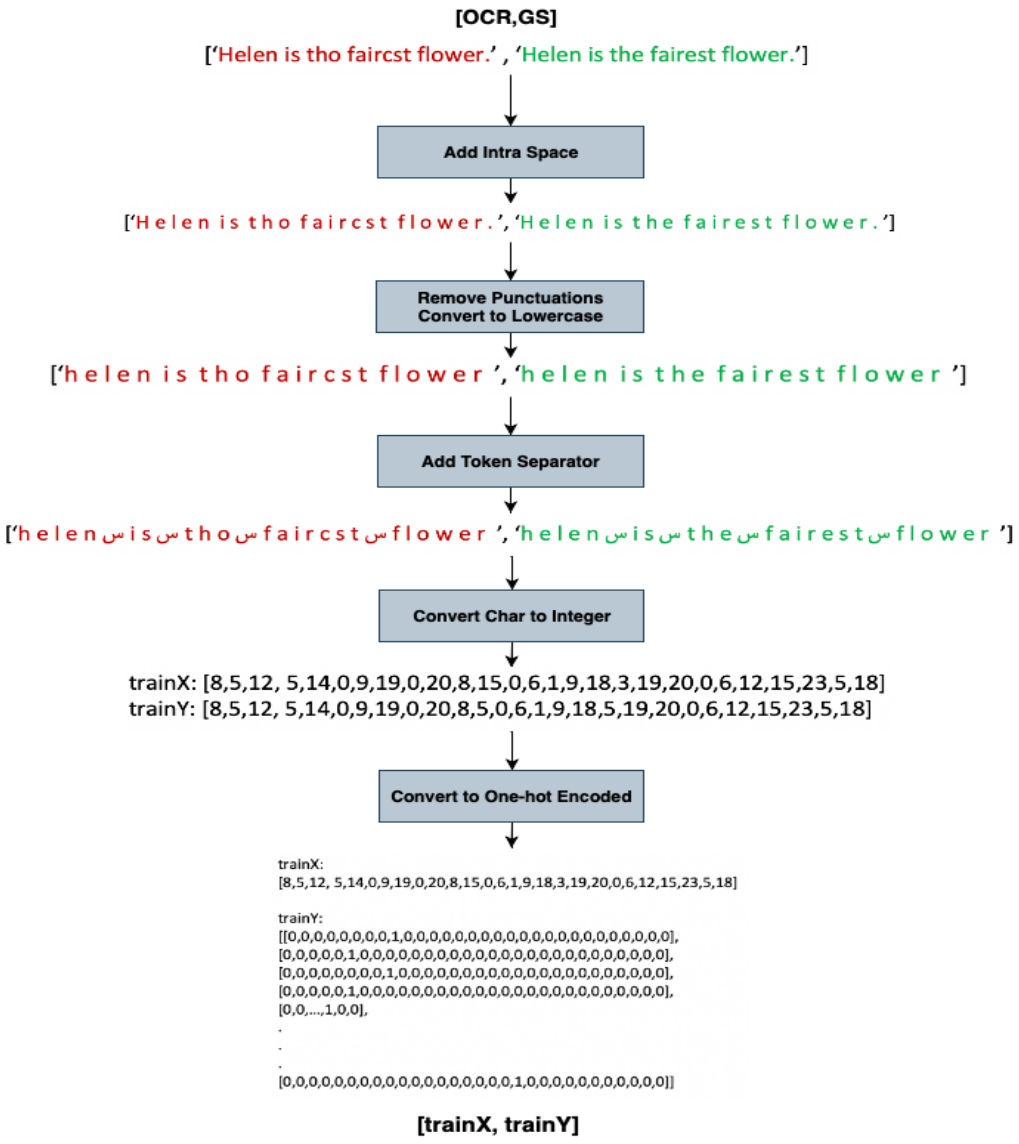



**Figure 3.4:** Entire Pre-process Steps on a Pair of OCR, GS

## 3.3 Neural Machine Translation Model

### 3.3.1 Model Architecture

#### 3.3.1.1 Algorithm Definition

Recurrent Neural Networks (RNN) use the idea of processing sequential information. The term recurrent illustrates that the same task is applied over each instance of the input such that the output is dependant on the previous computations and result. Tokens enter the recurrent units one by one and produce a fixed-size vector. RNNs have memory over previous computations and use this information in current processing. Therefore, they are able to capture the inherent sequential nature in the language where units are characters, words, or sentences and superior to convolutional neural networks in these domains. Furthermore, RNNs can handle inputs with arbitrary length.

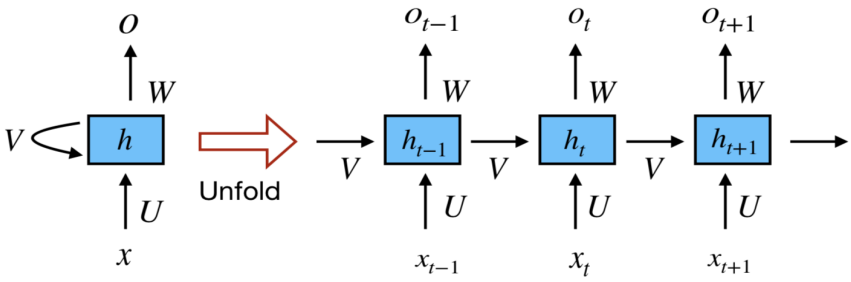



**Figure 3.5:** Simple Recurrent Neural Network [20]

Long Short-Term Memory (LSTM) has a unique mechanism to overcome the vanishing gradient problem which hampers the learning of long data sequences [30]. LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates. It consists of three gates: input, output and, forget gates.

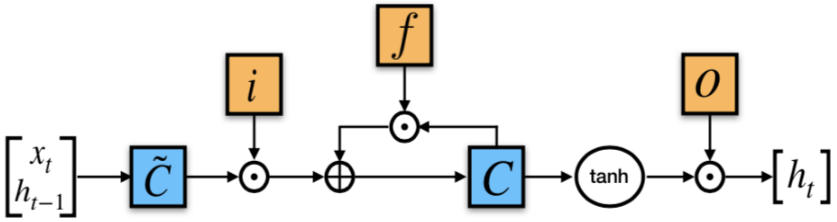



**Figure 3.6:** Long Short-Term Memory [9]

#### 3.3.1.2 Model Definition

First, it is necessary to specify the input and output of the model. Our Neural Machine Translation Model gets a fixed-length sequence as input and returns a one-hot encoded sequence of probabilities with the same length. Finding out the greatest probability for each character in a sequence, we can generate the output sequence.

Now, we define the architecture of our neural network. Our model is a encoder-decoder (seq2seq) LSTM model containing six layers.

- **Embedding Layer:** The first layer is an embedding layer due to the structure of the input, i.e. text. The Embedding layer is initialized with random weights and will learn an embedding for all of the characters in the training dataset. It gets three arguments as input: the size of the vocabulary, the size of the vector space in which characters will be embedded, and the input length, which is equal to the sequence length.

- **LSTM Layer:** The second layer is the Keras LSTM layer. This layer, along with embedding layer forms an encoder. The output of this layer is a fixed-size vector, a 2-dimensional matrix of outputs, that represents the internal representation of the input sequence.

- **RepeatVector Layer:** The RepeatVector layer follows the encoder part of the model. It repeats the output of LSTM layer equal to the length of the sequence and creates a 3D output. The RepeatVector layer can be used as an adapter to fit the encoder and decoder parts of the network together.

- **LSTM Layer:** Two LSTM layers besides the next layer form a decoder. We return the hidden sequence of LSTM layer rather than single values.

- **TimeDistributed Layer:** TimeDistributed layer is useful for many-to-many sequence prediction. We used TimeDistributed on the output layer to wrap a fully connected Dense layer with a single output to produce probabilities.

The activation function is softmax to predict the probability of each character in the output sequence. Additionally, the model is trained using the efficient Adam approach to stochastic gradient descent and minimizes the categorical loss function because we have framed the prediction problem as multi-class classification.

### 3.3.2 Model Training

Moving on training the model, we should split data into the training and evaluation set. For ICDAR 2017, this separation is predefined. The respective hyper-parameters of the models such as the number of epochs, batch size in addition to sequence length and step size are tuned experimentally according to the input.

After the first execution of the code, we encountered the memory limit problem on the GPU cluster. Although the dataset is small and fits in the memory, encoding input and output and making large matrices exceed the memory capacity. The solution is using fit_generator and data_generator. We can generate input and output on the fly and feed it right away to the deep learning model.

Now, it is time to train the model. The best model is saved for future use and prediction. Figure shows the schema of this step. We run different experiments and explain them at the end of this chapter. We should state that the model is trained specially for the error detection task.

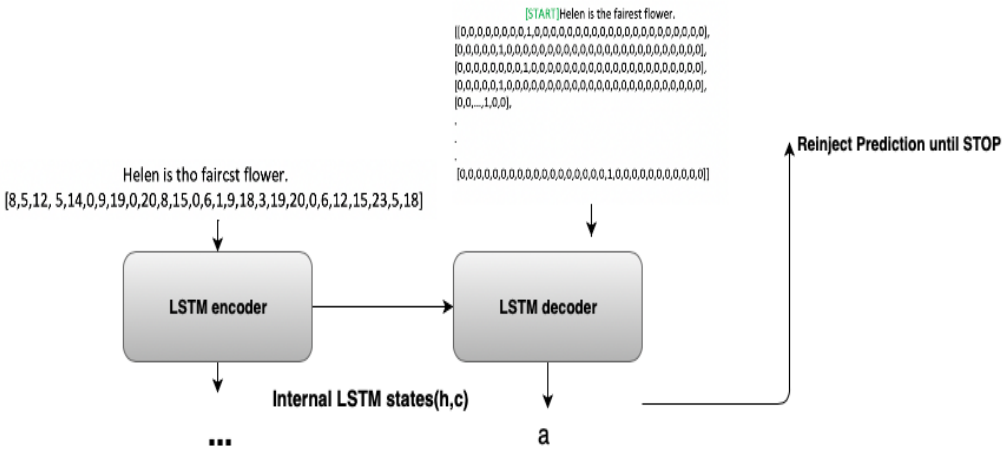



**Figure 3.7:** The Schema of Training the Seq2Seq Model

## 3.4 Post-processing

After training the model, the best model saved during training is utilized to predict the evaluation dataset. According to the previous section, the output is a one-hot encoded vector required to be converted to human-readable characters, Figure 3.8. For each character in the final output sequence, we select one with the highest probability to insert.

Afterward, we reinsert punctuations and uppercase characters based on the original OCR input.

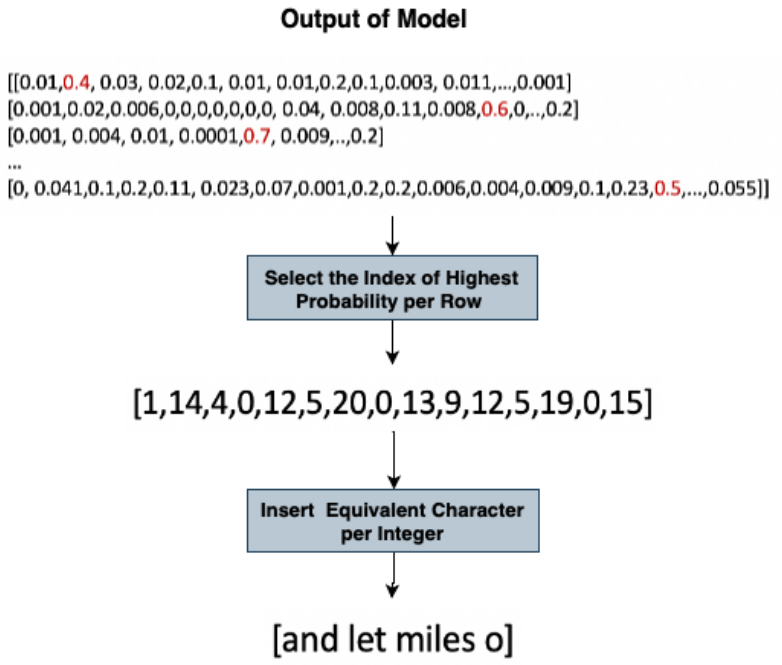



**Figure 3.8:** How to Convert the Model Output to a String: a Example

According to our alignment approach, one character in the OCR-ed text is predicted multiple times as our sliding windows overlap. Another issue arises to aggregate sequences of output and create the final document per OCR file. We did sub-sampling and discovered that the last time a particular character is predicted, it is more probable to get the corrected output from the model. In other words, the last sequence that the particular character is located in can produce the right prediction. We argue that this is because the model is fed of all relevant history and context beforehand and has a higher capability to learn. We also tested another aggregator which selects the prediction with the higher number of repetition (using majority). However, it falls behind the first aggregator. Figure 3.9 illustrates the components of the post-processing phase. Having the input and output sequences, we can move to model analysis section.

## 3.5 Experiments and Result Analysis

We ran three experiments with different configurations. We trained English monographs and periodicals separately. Then, We combined them and fed both into the model. The hy-

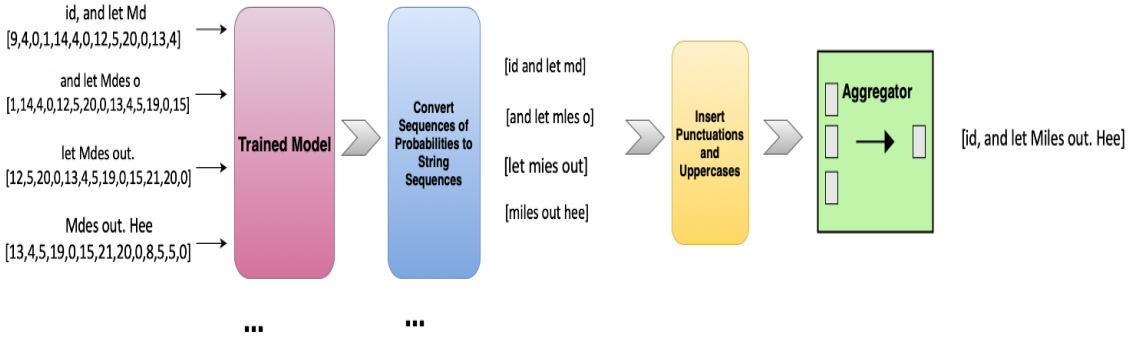



**Figure 3.9:** How to Convert Model Output to String: a Example

perparameter for all of them are identical and as follows: step_size=5, sequence_length=53, epoch=30. We keep the same model architecture for all three executions to be able to compare their results.

To evaluate the performance of the model, we can benefit from different metrics. Our primary metrics are precision, recall, and F1 calculated by the official script of ICDAR 2017 for the error detection task focusing on token level analysis. We also measure the internal outputs of the OCR post-correction system for better grasp of the model functionality. It is worth to mention that all the following performance statistics are macro-level averaging. We process each file separately and then average over all documents in proportion to the number of tokens they have at the end. Here, we report the final average.

### 3.5.1 Monograph

This model is trained only on monograph partition of the ICDAR 2017 dataset. We look into the results of the model from the token-level side using ICDAR 2017 evaluation script. Note that performance is translated in terms of precision, recall, and F1.

The output for this competition should contain the position of the erroneous tokens in addition to their length (how many tokens are involved in such error). The evaluation script quantify the precision, recall and, F1 for detection task per file. Table 3.1 summarize all measurement results.

**Table 3.1:** Performance of the System for English Monograph, ICDAR 2017

| Task | Precision | Recall | F1 |
|---|---|---|---|
| Detection | 0.1294 | 0.9226 | 0.2271 |

Table 3.1 proves the model has high recall. In short, more than 90% of errors are detected by the model. However, due to the high number of false positive cases, the model suffers from low precision and therefore, low F1. Similarly, micro-level analysis of internal data states that the strong point of the model is recall.

### 3.5.2 Periodical

Preparing an output file according to the format of the ICDAR 2017 competition, we evaluate the periodical model in terms of the detection task.

**Table 3.2:** Performance of the System for English Periodical, ICDAR 2017

| Task | Precision | Recall | F1 |
|---|---|---|---|
| Detection | 0.1609 | 0.9493 | 0.2689 |

We observe the same trend for periodicals. The 94% recall leads to detect almost all errors. The performance of the model trained on periodicals slightly exceed the model for monographs for both precision and recall.

### 3.5.3 Combination of Monograph and Periodical

The error rate for monograph is about 5% while it is 13% for periodicals. The combination of both partitions can balance the error rate. The model can learn a high rate of correct text besides more errors.

**Table 3.3:** Performance of the System on English Evaluation Set, ICDAR 2017

| Task | Precision | Recall | F1 |
|---|---|---|---|
| Monograph | 0.1398 | 0.9161 | 0.2368 |
| Periodical | 0.1593 | 0.9337 | 0.2655 |

Comparing the result of monographs, the model trained on monograph documents slightly exceeds the one trained on all English data (monographs and periodicals). We argue that since the error types of monographs are different from periodicals, combining monographs and periodicals couldn't help the model effectively learn the data.

Moving on to periodicals, we can conclude that increasing the volume of training data doesn't have a significant effect on the performance of the model, as we see 1% degradation of performance.

Setting the results of monographs and periodicals side by side, we see that the model outperforms for periodicals to some degree. However, top-level analysis asserts that there is no remarkable difference between the models trained on each genre separately and one trained on their combination.

As we mentioned earlier, all models perform ideally in terms of recall, and they can find more than 92% of errors. One solution to improve the model is adding a human assistant within the pipeline analogous to the semi-automated system of ICDAR competition. In this configuration, we show the position of error and list of candidates including the model-generated output and also the original OCR-ed input to the assistant. The human assistant can decide to substitute the model output or continue with no change option. Due to the high recall of the system, we can correct the text almost fully. Again, we extracted token-level statistics to quantify the effect of the semi-automated system on the OCR post-correction.

**Table 3.4:** Top-5 Erroneous Tokens in Evaluation Set After Post-Correction, ICDAR 2017

| Monograph | | Periodical | |
|---|---|---|---|
| OCR=>GS | Occurrence | OCR=>GS | Occurrence |
| re=>re-ceived | 4 | .=>c., | 11 |
| be=>be. | 3 | and=>land, | 6 |
| Gen=>Gen-tleman | 3 | at=>flat, | 4 |
| to=>to-gether | 3 | A=>A. | 3 |
| con=>con-tended | 2 | day=>Saturday | 2 |

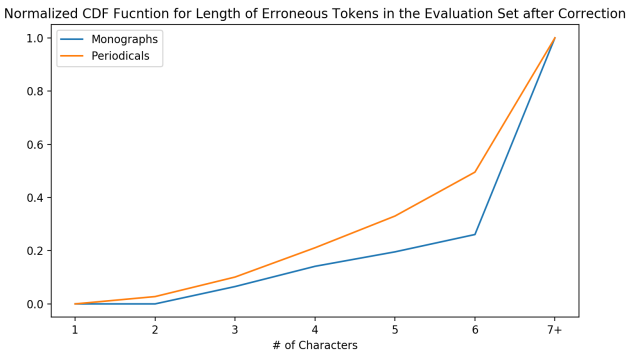
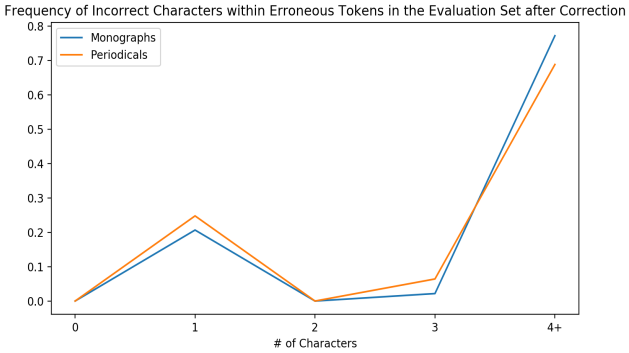



**Figure 3.10:** Length of Erroneous Tokens After Correction, ICDAR 2017

After making our pipeline semi-automated, we can observe remarkable improvement in correcting the erroneous documents. Now, the total number of incorrect tokens are 92

**Figure 3.11:** Number of Incorrect Characters within Erroneous Tokens After Correction, ICDAR 2017




and 109 in monographs and periodicals, respectively. We achieved to correct nearly all words with 1-6 characters for monographs and 1-3 characters for periodicals. Moreover, the words with less than 4 faulty characters are revised in both sections of the evaluation set, as we've expected. We fixed almost all of word segmentation errors and tokens with 2-3 incorrect characters. Furthermore, most of the tokens with a single character error were corrected too, with the overall performance of 95% for both tasks combined.

## 3.6 Conclusion

In this chapter, we studied the second research question: *What is the performance of the seq2seq model on OCR post-correction?* We designed a character-level seq2seq model. The model achieves recall more than 92% for the error detection task, which is our **first** finding. This outcome motivated us to include the human-assistant in the pipeline to fix the flagged errors. Evaluation of the semi-automated system claims about 95% for F1 to detect and correct error concurrently.

Setting various experiments on different genres of the dataset, as a **second** finding, we figured out the performance does not change. Even the model trained on the combination of both genres performs similarly to two individual models trained on monographs and periodicals.

As a **third** finding, we found that even with the model capable of detecting all errors, including a human in the loop is unavoidable.

# Chapter 4

# ICDAR 2019 Competition

In this chapter, we are going to study the third research question: *With limited training data, how can we take advantage of pre-trained models and transfer learning?* At the end of April 2019, we submitted the output of our seq2seq model for ICDAR 2019 competition. In this chapter, we examine different solutions such as fine-tuning and transfer learning to tackle the lack of sufficient data for English section of ICDAR 2019.

## 4.1 ICDAR 2019

ICDAR 2019 dataset contains 20 million characters (3.5M token) from different sources and 10 European languages (English, French, German, Finish, Spanish, Dutch, Czech, Bulgarian, Slovak, and Polish) with the unbalanced distribution. Each language includes one or several sub-folders. ICDAR 2019 is richer than ICDAR 2017 in terms of the number of characters and languages. The structure of each text files is similar to the previous edition.

| | | | | Nb of character | | |
|---|---|---|---|---|---|---|
| ID | Language | Subset | Nb of file | Total | Trainset | Testset |
| 1 | BG | BG1 | 150 | 241 552 | 239 136 | 2 416 |
| 2 | CZ | CZ1 | 150 | 210 382 | 208 278 | 2 104 |
| 3 | DE | DE1 | 102 | 575 416 | 460 333 | 115 083 |
| | | DE2 | 150 | 387 466 | 309 973 | 77 493 |
| | | DE3 | 7623 | 10 018 258 | 8 014 606 | 2 003 652 |
| | | DE4 | 321 | 509 757 | 407 806 | 101 951 |
| | | DE5 | 654 | 818 711 | 654 969 | 163 742 |
| | | DE6 | 773 | 935 014 | 748 011 | 187 003 |
| | | DE7 | 415 | 527 845 | 422 276 | 105 569 |
| 4 | EN | EN1 | 150 | 190 995 | 189 085 | 1 910 |
| 5 | ES | ES1 | 150 | 406 365 | 402 301 | 4 064 |
| 6 | FI | FI1 | 393 | 1 960 345 | 1 568 276 | 392 069 |
| 7 | FR | FR1 | 1172 | 2 792 067 | 2 233 654 | 558 413 |
| | | FR2 | 150 | 174 050 | 170 569 | 3 481 |
| | | FR3 | 1968 | 742 574 | 594 059 | 148 515 |
| 8 | NL | NL1 | 150 | 586 524 | 580 659 | 5 865 |
| 9 | PL | PL1 | 150 | 231 937 | 229 618 | 2 319 |
| 10 | SL | SL1 | 150 | 180 401 | 178 597 | 1 804 |
| **TOTAL** | **10** | **18** | **14771** | **21 489 659** | **17 612 206** | **3 877 453** |

**Figure 4.1:** Details of ICDAR 2019 Dataset

At the time of writing the current thesis, the training and evaluation sets are only sent to the participants and not published publicly. Furthermore, testing documents only have OCR-ed input (no aligned OCR and no aligned GS). Although our model is language-independent, we focus on the English partition to take advantage of dataset released for 2017. The English folder contains one sub-folder with 148 documents and 243107 characters.

## 4.2 ICDAR 2019 OCR Error Analysis

Analogous to Chapter 2, we obtain statistics from token level and character level sides only for English folder.

### 4.2.1 Statistics on Token Level

We split the tokens by white space to find out their numbers, the systematic errors, the length of erroneous tokens and, the number of faulty character within each of them. Table 4.1 depicts an overview of the number of existing tokens in the training and evaluation set. Accordingly, the error rate in the training set is 35%. Not only the size of the training data is small, but also the error rate is considerably high.

**Table 4.1:** Number of Tokens in English Set, ICDAR 2019

| Number of Tokens | Training | Testing |
|---|---|---|
| Ground Truth | 37704 | NA |
| Original OCR | 37896 | 11385 |
| Erroneous | 13331 | NA |

In Table 4.2, we focus on the systematic errors. The frequency of such errors acknowledges that most of them are distributed in the tale of the graph, making the detection and correction task much more difficult. Accordingly, the sum of their frequencies is less than 3%. At a glance, some of the erroneous tokens seem meaningless since there is no logical relationship between the mapping (OCR=>GS). This is because of the structure of documents. In many of them, we see many differences between the OCR-ed text and the respective GS. Each even contain many sentences that don't have equivalents in the other. As a result, the word alignment becomes unreliable. Furthermore, we can see that some errors are related to punctuations. The remaining are classified in the group of character

format which happens due to the similarity of their character glyph such as tbe=>the. There are also some similarities to ICDAR 2017, for instance, tokens 1=>I, tbe=>the

**Table 4.2:** Top-10 Erroneous Tokens in English Training Set, ICDAR 2019

| English 2019 | | |
|---|---|---|
| OCR=>GS | Occurrence | Freq.(%) |
| the=>who | 84 | 0.6 |
| of=>he | 50 | 0.3 |
| and=>the | 38 | 0.3 |
| 1=>I | 34 | 0.3 |
| tbe=>the | 26 | 0.2 |
| ano=>and | 20 | 0.15 |
| .=>; | 19 | 0.15 |
| a=>an | 16 | 0.1 |
| >=>? | 14 | 0.1 |
| fame=>same | 13 | 0.1 |

We also look into the length of incorrect tokens and the number of misspelled characters within. If a token has more characters and the number of faulty ones is less relatively, it is more probable that the model can correct it. Based on Figure 4.2, the length of the erroneous tokens is approximately distributed uniformly for characters from 1 to 6. However, tokens with 7 and more character constitute the greater number among all. Figure 4.3 reports the number of incorrect characters withing tokens. The tokens with one faulty character have the greatest number. In comparison with ICDAR 2017, there are fewer tokens with a single error character while the number of tokens composed of more than 3 incorrect characters is raised.

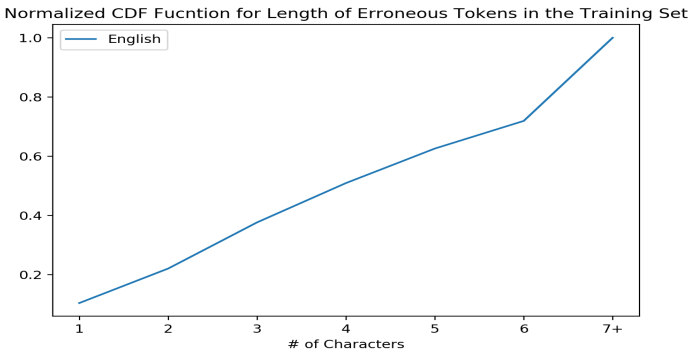



**Figure 4.2:** Length of Erroneous Tokens in English Training Set, ICDAR 2019

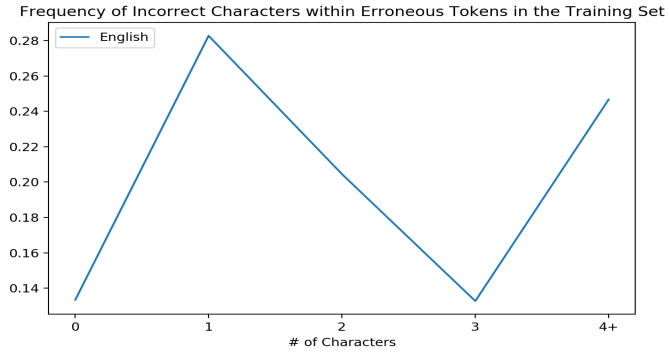



**Figure 4.3:** Number of Incorrect Characters within Erroneous Tokens in English Training Set, ICDAR 2019

### 4.2.2 Statistics on Character Level

In this section, we analyze the dataset from the character point of view. Having OCR-ed text and GS aligned, we count the number of characters, systematic character error (unigrams and bigrams), and also the rate of insertion, deletion, substitution required to reach GS from OCR document.

With Table 4.3, we get insight into the number of characters both in training and testing dataset. Accordingly, the character error rate is 24%.

**Table 4.3:** Number of Characters in English Set, ICDAR 2019

| Number of Characters | training | testing |
|---|---|---|
| Ground Truth | 207778 | NA |
| Original OCR Input | 209311 | 62825 |
| Erroneous | 50455 | NA |

Moving on systematic errors, Table 4.4 presents the top-10 character errors for unigrams. We skip bigrams, trigrams, and more because their frequency is negligible.

Even the occurrence of such common errors is low (less than 2%) relative to the total number of faulty characters. The more white error in our dataset, the harder OCR post-correction is. We can group the errors into punctuation errors and character format such as [c,e], [o,d], [e,t]. The systematic errors in ICDAR 2019 and ICDAR 2017 do not have features in common.

**Table 4.4:** Top-7 Unigram Errors in English Training Set, ICDAR 2019

| English 2019 | | |
|---|---|---|
| OCR=>GS | Occurrence | Freq.(%) |
| c=>e | 358 | 0.7 |
| ,=>’ | 267 | 0.5 |
| o=>d | 169 | 0.3 |
| i=>u | 40 | <0.1 |
| d=>l | 31 | <0.1 |
| )=>h | 29 | <0.1 |
| e=>t | 27 | <0.1 |

As the final table, we point out Figure 4.4, which calculated the number of insertion, correction, deletion essential to correct OCR-ed text. The distribution of those three actions is approximately uniform, which is in contrast with ICDAR 2017 with many substitutions.

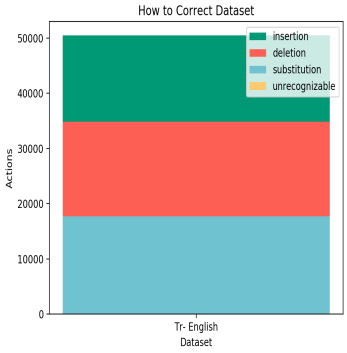



**Figure 4.4:** Insertion, Deletion, Substitution Required to Correct English Training Set, ICDAR 2019

To avoid repetition, we need not to explain data pre-processing, the model architecture, data post-processing steps. For more information, please refer to Chapter 3.

## 4.3 Model Training

Before training any model, we should specify the training and testing set. This step is straightforward for us because it is predefined. However, an issue arises. Although in the previous chapter, we had more training data, the size of data was not sufficient. The size of the English partition of ICDAR 2019 is about 100 times smaller than ICDAR 2017. Furthermore, by breaking down the training data into the validation set, we lose 20% of such small data. Since about 25% of characters are erroneous (which is too high) the

model is not able to differentiate between correct and faulty characters. In other words, the high rate of error makes the situation troublesome for LSTM to generalize. Therefore, we decided to benefit from the models trained on English documents of ICDAR 2017. In addition to transfer learning, we benefit from fine-tuning the English monograph and combination of English monograph and periodicals on ICDAR 2019 with 5-fold cross-validation.

## 4.4 ICDAR 2019 Output Format

The output of ICDAR 2019 must illustrate the position of erroneous tokens in addition to their length, Figure 4.5. To find out the positions of faulty characters, we compare the output derived from post-processing and the original OCR-ed text. We iterate over original OCR-ed text to align tokens with output and discover erroneous ones.

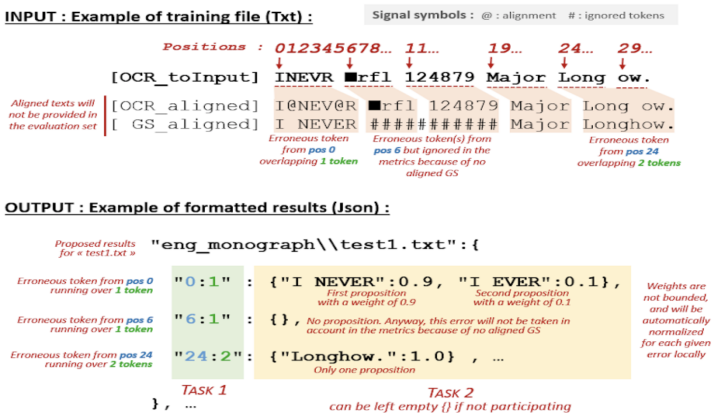



**Figure 4.5:** Tasks of ICDAR Competition: Example of Input and Output

## 4.5 Experiments and Results Analysis

As we discussed earlier, we set up several experiments. The first model was trained and tested on ICDAR 2019 - considered as baseline. In the the next experiment, we carried out transfer learning and made use of the pre-trained model of ICDAR 2017. The third model was built upon all English documents including monographs and periodicals of ICDAR 2017. The last one was fine-tuned on monographs of ICDAR 2017. The value of hyperparameters was tuned per experiment to maximize the performance. Since the corresponding GS of the testing data is not published yet, we used another metric to

assess the model. The benchmark for evaluating and comparing these three models are the validation set.

1. **Training on English Files 2019**

   Setting hyperparameters as follows: sequence_length=40, n_cells=265, step_size=4 and, n_layers=5, our baseline worked extremely unsatisfactory. The final accuracy is 0.28, and it is 0.27 for validation accuracy. Neither the model could learn the training data, nor it could generalize what it has learned yet.

2. **Pre-training on English Files (Monographs and Periodicals) 2017**

   In this experiment, we utilized the model of Chapter 3. Without doing extra training, we adjusted the model for this use case. The characteristics of the model is as follows: sequence_length=53, n_cells=512, step_size=5 and, n_layers=6.

3. **Fine-tuning the Model of English Files (Monographs and Periodicals) 2017**

   In similar fashion to the previous experiment, it is required to be faithful to the configuration of pre-trained model with following setup: sequence_length=53, n_cells=512, step_size=5 and, n_layers=6. The maximum accuracy and validation accuracy are 0.37 and 0.35 respectively.

4. **Fine-tuning the Model of English Monographs 2017**

   Having the Pre-trained model on monograph 2017, we fed the model more data with a lower rate of error. The value of hyperparameters are identical to the model we trained for Chapter 3, sequence_length=53, n_cells=512, step_size=5 and, n_layers=6. Otherwise it wouldn't be possible to take advantage of the weights and apply fine-tuning. The results were 0.51 and 0.57 after fine-tuning for accuracy and validation accuracy respectively.

Table 4.5 summarizes the performance for all experiments.

**Table 4.5:** Result of Experiments for ICDAR 2019 Competition

| Experiment | Accuracy | Validation Accuracy |
|---|---|---|
| en19 | 0.28 | 0.27 |
| en17 | 0.50 | 0.49 |
| en17_en19_ft | 0.37 | 0.35 |
| mono17_en19_ft | 0.51 | 0.53 |

According to Table 4.5, the least score is for our baseline, which is the first experiment. The small quantity of ICDAR 2019 for English besides the high rate of errors justify the weakness of this model. Experiments demonstrate that making use of the model trained on English ICDAR 2017 is beneficial with more than 20% growth in the accuracy. We went one step further to improve the performance by fine-tuning such a model. However, the accuracy dropped. We repeated fine-tuning on a genre-specific model (in our case, monographs) and gained the best score. We can argue that the sources of English documents in 2019 are more analogous to the monographs of 2017 rather than periodicals. It seems that error patterns in periodicals 2017 are far from ones in the ICDAR 2019. The format of monographs is homogeneous while each periodical may have its unique style and consequently specific error types. Thereby, the fourth model achieves the highest accuracy even superior to transfer learning. Using the fourth model, we generated the output for the ICDAR 2019 competition and submitted it.

## 4.6 Conclusion

In this chapter, we concentrated on the third research question: *With limited training data, how can we take advantage of pre-trained models and transfer learning?* We aimed to find out how we can benefit from different approaches such as transfer learning and fine-tuning to boost the model performance. We had ICDAR 2019 English documents as our core dataset. After running several experiments, we formulated our findings into three statements. **First**, we figured out transfer learning has a positive effect on the model improvement. **Second**, fine-tuning could be practical. The impact of fine-tuning is dependent on our setting and the closeness of the data used for pre-training and one for fine-tuning. **Third**, due to the fact that transfer learning outcomes are relatively high, there is no need to spend much effort, time, and money to carry out fine-tuning. We can aggregate a rich corpus, train a model once, and make use of the pre-trained model in different use cases.

# Chapter 5

# Multilingual Transfer Learning

We address the fourth research question in this chapter: *Can we even transfer OCR post-correction models across languages?* We plan to run more experiments to make our system bilingual. In the previous chapter, we noticed that applying transfer learning is beneficial. Going one step forward, we would like to figure out to what extent our models are language-blind and how we can adjust them to the new environment via fine-tuning.

## 5.1 Motivation

Both the dataset of ICDAR 2017 and 2019 contains documents from various European languages. The volume of data for some languages such as English in 2019 seems insufficient for model training. It would be better to train one robust model on top of all languages instead of merging multiple weak models per language. The OCR errors are mostly matter of the characters rather than being exclusive for a specific language. Furthermore, the nature of our seq2seq approach is language-independent allowing us to adapt it for various languages.

## 5.2 French ICDAR 2017

We make use of all partitions in ICDAR 2017 dataset for experiments of the current chapter:

- eng_monograph
- fr_monograph
- fr_periodical
- eng_periodical

The complete analysis of English files can be found in Chapter 2. To avoid repetition, in this section, we only look into statistics of French set from character and token level points of view.

Analogous to the English section, French set consists of monograph and periodical. The training set has 977 files for monographs and 85 for periodicals. The number of documents in the evaluation set is 108 and 23, respectively.

## 5.3 OCR Error Analysis of French ICDAR 2017

### 5.3.1 Statistics on Token Level

In this section, we count the number of tokens for each partition of French set for both the training and evaluation data. Furthermore, similar to Chapter 2, the length of erroneous tokens and the number of incorrect characters within errors will be reported here.

Based on Tables 5.1 and 5.2, error rate in the training set is 7% and 14% for monographs and periodicals, respectively; while it is 6% and 8% in the evaluation set. There is a remarkable gap between the periodicals in the training set and evaluation set in terms of the error rate. The statistics demonstrate that the token error rate for English and French are approximately the same in the training set.

**Table 5.1:** Number of Tokens in French Training Set, ICDAR 2017

| Number of Tokens | Monograph | Periodical |
|---|---|---|
| Ground Truth | 598944 | 268829 |
| Original OCR | 605458 | 255612 |
| Erroneous | 44287 | 37580 |

**Table 5.2:** Number of Tokens in French Evaluation Set, ICDAR 2017

| Number of Tokens | Monograph | Periodical |
|---|---|---|
| Ground Truth | 61794 | 101828 |
| Original OCR | 63035 | 104866 |
| Erroneous | 3894 | 9331 |

Looking into Tables 5.3 and 5.4, we can observe the systematic errors in the training and evaluation set. Analogous to English documents, the occurrence of such errors is infrequent,

even less that 3%. The pattern of systematic errors for French files differs from the English set. It is reasonable since they are different languages with different vocabularies.

**Table 5.3:** Top-10 Erroneous Tokens in French Training Set, ICDAR 2017

| Monograph | | | Periodical | | |
|---|---|---|---|---|---|
| OCR=>GS | Occurrence | Freq.(%) | OCR=>GS | Occurrence | Freq.(%) |
| font=>sont | 468 | 1 | 1=>. | 138 | 0.3 |
| a=>sa | 136 | 0.3 | ta=>la | 134 | 0.3 |
| ie=>de | 136 | 0.3 | do=>de | 134 | 0.3 |
| vn=>un | 116 | 0.2 | tes=>les | 100 | 0.2 |
| le=>lle | 98 | 0.2 | h=>à | 91 | 0.2 |
| foit=>soit | 87 | 0.2 | la=>Ia | 78 | 0.2 |
| !.=>!... | 79 | 0.1 | Il=>II | 76 | 0.2 |
| fous=>sous | 68 | 0.1 | te=>le | 75 | 0.2 |
| vne=>une | 55 | 0.1 | La=>-La | 75 | 0.2 |
| i=>1 | 54 | 0.1 | le=>Ie | 73 | 0.2 |

**Table 5.4:** Top-10 Erroneous Tokens in French Evaluation Set, ICDAR 2017

| Monograph | | | Periodical | | |
|---|---|---|---|---|---|
| OCR=>GS | Occurrence | Freq.(%) | OCR=>GS | Occurrence | Freq.(%) |
| 1=>2 | 67 | 1.7 | do=>de | 43 | 0.4 |
| pou-ces=>pou,ces | 28 | 0.7 | 1=>! | 41 | 0.4 |
| fous=>sous | 21 | 0.5 | a=>à | 34 | 0.3 |
| I=>1 | 16 | 0.4 | lu=>du | 27 | 0.3 |
| z=>2 | 15 | 0.4 | el=>et | 25 | 0.2 |
| 11=>12 | 14 | 0.4 | la=>là | 24 | 0.2 |
| i=>I | 11 | 0.3 | 11=>Il | 20 | 0.2 |
| da=>du | 9 | 0.2 | le=>la | 20 | 0.2 |
| II=>11 | 9 | 0.2 | Je=>de | 15 | 0.1 |
| a=>à | 8 | 0.2 | ia=>la | 15 | 0.1 |

Figures report the length of incorrect tokens, 5.2 and 5.3. In all sections, the number of tokens with 7 or more characters is standing at the top, like the English set. In monographs, the rest of the tokens uniformly distributed between other categories (1-6 characters). However, tokens with 2-3 characters are higher than others in periodicals. According to these figures, we can assume that all monographs of ICDAR 2017 are identical nearly regardless of their languages. The same statement is also valid for periodicals.

**Figure 5.1:** Length of Erroneous Tokens in French Set, ICDAR 2017

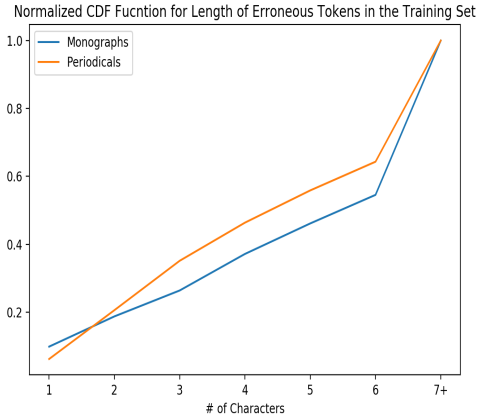



**Figure 5.2:** Training Set

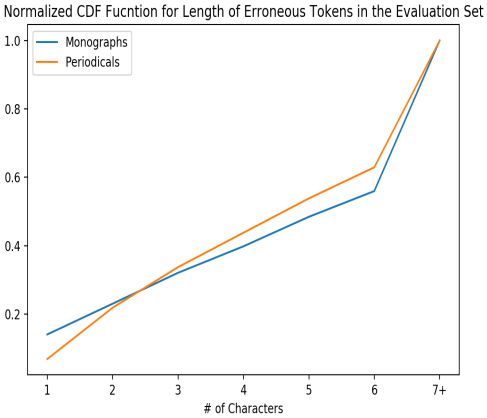



**Figure 5.3:** Evaluation Set

We also remark that although tokens with single character error constitute the majority, Figures 5.5 and 5.6, they are not as frequent as tokens with same condition in the English set. We also observe farther word segmentation errors in the French section rather than in the English ones.

**Figure 5.4:** Number of Incorrect Characters within Erroneous Tokens in French Set, ICDAR 2017

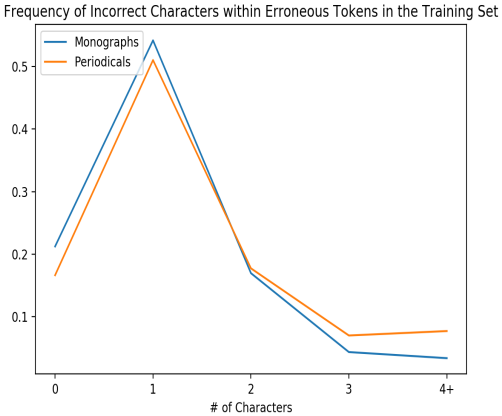



**Figure 5.5:** Training Set

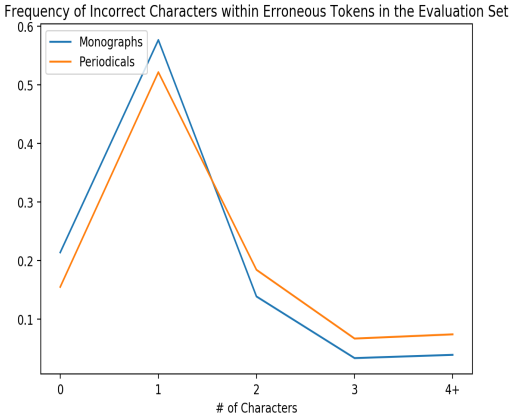



**Figure 5.6:** Evaluation Set

### 5.3.2 Statistics on Character Level

Sticking to the structure of the current thesis, we provide the information regarding the number of characters, systematic errors, number of insertion, deletion, and substitution in this section.

From the character sides, the error rate is 2% and 8% in monographs and periodicals of the training data. The measurements for the evaluation set is 2% in monographs and 4% in periodicals , as shown in Tables 5.5 and 5.6. Following the same pattern in token stats, for periodicals, the rate of errors in the training set is two times greater than the evaluation set. In general, error rates in the French partition of ICDAR 2017 are fewer in contrast to English documents in both genres.

**Table 5.5:** Number of Characters in French Training Set, ICDAR 2017

| Number of Characters | Monograph | Periodical |
|---|---|---|
| Ground Truth | 3560503 | 1637088 |
| Original OCR | 3569322 | 1640239 |
| Erroneous | 74044 | 142846 |

**Table 5.6:** Number of Characters in French Evaluation Set, ICDAR 2017

| Number of Characters | Monograph | Periodical |
|---|---|---|
| Ground Truth | 358707 | 630165 |
| Original OCR Input | 360692 | 632523 |
| Erroneous | 6806 | 25896 |

In tables for systematic errors ( Tables 5.7 and 5.8), we can see punctuation errors, errors caused by character format error, and diacritics with downward frequencies. Accordingly, French and English have many character errors in common.

**Table 5.7:** Top-10 Character Errors in French Training Set, ICDAR 2017

| Monograph | | | Periodical | | |
|---|---|---|---|---|---|
| OCR=>GS | Occurrence | Freq.(%) | OCR=>GS | Occurrence | Freq.(%) |
| f=>s | 3473 | 4.6 | t=>l | 792 | 0.5 |
| l=>t | 818 | 1.1 | l=>i | 360 | 0.2 |
| è=>e | 238 | 0.3 | a=>s | 324 | 0.2 |
| ,=>. | 212 | 0.3 | e=>c | 272 | 0.2 |
| u=>n | 200 | 0.2 | i=>t | 212 | 0.1 |
| n=>m | 181 | 0.2 | à=>a | 134 | <0.1 |
| i=>1 | 70 | <0.1 | n=>r | 92 | <0.1 |
| E=>É | 57 | <0.1 | I=>1 | 57 | <0.1 |
| h=>b | 44 | <0.1 | J=>j | 47 | <0.1 |
| x=>X | 43 | <0.1 | S=>3 | 29 | <0.1 |

**Table 5.8:** Top-10 Character Errors in French Evaluation Set, ICDAR 2017

| Monograph | | | Periodical | | |
|---|---|---|---|---|---|
| OCR=>GS | Occurrence | Freq.(%) | OCR=>GS | Occurrence | Freq.(%) |
| é=>e | 25 | 0.3 | i=>a | 49 | 0.2 |
| ,=>. | 17 | 0.2 | î=>i | 42 | 0.1 |
| e>c | 15 | 0.2 | o=>c | 27 | 0.1 |
| i=>1 | 12 | 0.1 | 1=>! | 26 | 0.1 |
| y=>v | 10 | 0.1 | è=>é | 26 | 0.1 |
| E=>É | 9 | 0.1 | E=>É | 24 | <0.1 |
| ù=>û | 8 | 0.1 | n=>r | 14 | <0.1 |
| l=>f | 7 | 0.1 | 3=>8 | 13 | <0.1 |
| n=>i | 7 | 0.1 | a=>d | 13 | <0.1 |
| h=>n | 6 | <0.1 | 0=>6 | 11 | <0.1 |

We also keep track of the numbers of insertion, deletion, and substitution required to convert OCR-ed files into GS documents, as seen in Table 5.7. In the training set, substitution has the most considerable amount for monographs; while it is in the second order for periodicals (after unrecognizable characters). There are nmany instances of unrecognizable characters in periodicals due to the difficulty of the OCR system for distinguishing text in different layouts. Dropping unrecognizable characters, substitution and deletion constitute the majority in all cases even regardless of language.

Note that the fewer number of unrecognizable characters in French strengthen the claim that the OCR system is customized for French rather than English, which led to the higher quality of dataset for French.

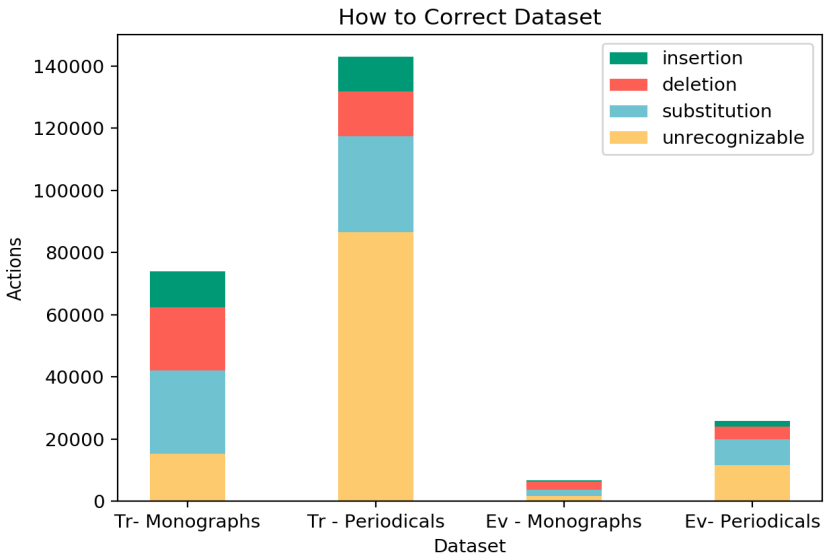



**Figure 5.7:** Actions Required to Correct French Documents

We skip the Pre-processing, Model Architecture, and Post-processing sections to avoid repetition. The elaborate explanation of these parts can be found in Chapter 3.

## 5.4 Model Training

We ran several experiments to examine the impact of transfer learning and fine-tuning in a bilingual environment. To do so, we specified the training data in line with our setup. The evaluation data is fixed for all cases, which is the English training data of ICDAR 2019. The rationale behind selecting the training data of English ICDAR 2019 is that the corresponding evaluation set is not published yet. Therefore, if we pick it for testing our models, we can't assess the outcomes.

- **Training on French**

  The training step is identical to what we have done in chapter 3. We trained three models on monographs, periodicals, the entire French set, respectively.

- **Pre-training on French, Fine-tuning on English**

  We broke it down into three executions, pre-training on French monographs, pre-training on French periodicals, and pre-training on the entire French set.

The rationale behind first training on French and then execute fine-tuning is that the French alphabet maps to the English keyboard, though it has diacritics.

English and French are linguistically independent languages. To fill the gap in between, we require to move from French to English smoothly. The model first learns French and then gradually learns English (not in one go). To do so, this process must be carried out in different runs. The spectrum can be like this:

100% French, 0% English => 80% French, 20% English => 60% French, 40% English

Due to the time limitation, we skip a level in between and in the fine-tuning step. Furthermore, we shuffle files before feeding them into the model.

## 5.5 Experiments and Result Analysis

Here, there is a list of all experiments. We keep the model architecture consistent in order to be comparable.

- **EXP1: Pre-training on English Monographs 2017**
- **EXP2: Pre-training on English Periodicals 2017**
- **EXP3: Pre-training on English Set (Monographs and Periodicals) 2017**
- **EXP4: Pre-training on French Monographs 2017**
- **EXP5: Pre-training on French Periodicals 2017**
- **EXP6: Pre-training on French Set (Monographs and Periodicals) 2017**
- **EXP7: Fine-tuning the Model of French Monographs 2017 with French Monographs 2017**
- **EXP8: Fine-tuning the Model of French Periodicals 2017 with French Periodicals 2017**
- **EXP9: Fine-tuning the Model of French Set (Monographs and Periodicals) 2017 with English Set 2017**

**Table 5.9:** Performance of the System for in Cross-lingual Environment

| Experiment | Precision | Recall | F1 |
|---|---|---|---|
| EXP1 | 0.3141 | 0.9219 | 0.4685 |
| EXP2 | 0.2682 | 0.9688 | 0.4201 |
| EXP3 | 0.4524 | 0.7373 | 0.5607 |
| EXP4 | 0.3367 | 0.8826 | 0.4874 |
| EXP5 | 0.3538 | 0.8237 | 0.4949 |
| EXP6 | 0.3475 | 0.8304 | 0.4899 |
| EXP7 | 0.3734 | 0.8293 | 0.5149 |
| EXP8 | 0.3999 | 0.7678 | 0.5258 |
| EXP9 | 0.4290 | 0.7590 | 0.5481 |

The First three experiments assess the models in Chapter 3 on the new dataset, ICDAR 2019. The model for periodicals has the least performance among all. Again, we can argue that error patterns in periodicals are far from the errors in standard files, and the sources of ICDAR 2019 could be more analogous to monographs. On the other hand, the model trained on the entire English outperforms the model for monographs. Not only the volume of training data increments but also the error rate in the combination is balanced for experiment 3.

We evaluated the impact of transfer learning in experiments 4-6. The performance of those three models is almost similar. Comparing each experiment with its equivalent in English set (EXP4 with EXP1, EXP5 with EXP2), we observe that the French model has less recall and higher precision. Since the French model doesn't specifically learn English errors, flagging fewer number of errors is reasonable. By reducing the number of false positive cases, French models achieve higher precision. There is an embedded post-correction step within the OCR system customized per language. We are not aware of the structure of this top layer, and it could conflict with our English models. With the help of transfer learning, we eliminate the effect of such an embedded layer. As a result, we obtain higher precision for French models.

Experiment 7-9 outcomes illustrate the superiority of fine-tuning to transfer learning. The model learns both French and English and gains more precision. Comparing the bilingual models with English models ( EXP7 and EXP1, EXP 8 and EXP2), we can see the dominance of bilingual models in terms of precision and F1. Moreover, setting experiment 3 and experiment 9 side by side, we can see the performance of English model and French-English model are almost identical. Note that the proportion of English files

in fine-tuning in experiment 7-9 is one-third. In othe words, the majority of training data in the bilingual model is French while its performance is analogous to the English model.

According to the results, since our character-level seq2seq model is language-blind, we can benefit from the entire training data regardless of their language by means of fine-tuning. We must start from one language, smoothly step forward to the other one to cover the gap between languages.

## 5.6 Conclusion

In this chapter, we studied the fourth research question *Can we even transfer OCR post-correction models across languages?* Our corpus consists of various languages. Due to the independence of our model on specific languages, we proposed to build up a model making use of all languages instead of one model per language. We tested our solution only in French and English. However, it could be expanded to other languages as well.

Centering on our findings, **first**, we figure out transferring one model to another language leads to higher precision and less recall. Although the overall performance (F1) is beyond the model which particularly trained on the language, we can still benefit from this approach.

Our experiments proved the advantage of the fine-tuning approach. As the **second** finding, we claimed that there is no demand to have sufficient training data per language. Developing one model on top of all languages can gain comparable performance.

**Thirdly**, we found that OCR errors are independent of a specific language. Not only we can enrich our training data across languages but also we achieve training a model on top all of the languages with high performance.

# Chapter 6

# Impact on Standard NLP Tasks

In this chapter, we investigate the fifth research question: *How do improvement in OCR quality impact downstream NLP tasks, such POS-tagging and NER?*. We would like to know to what extent the improvement in the quality of OCR-ed documents will be translated in the outcome of standard NLP tasks.

## 6.1 Motivation

The OCR systems can form the first stage in a pipeline where later stages assigned to complicated tasks in various applications of information retrieval such as question answering, summarization, topic detection and, inquiry in search engines and so on [21]. So, it is essential to understand the influence of error detection and post-correction of OCR outputs on text analysis routines. The errors in the early stages will be propagated to the end of the pipeline and degrade the performance of final output exponentially. In addition to enhancing the post-correction system, it is worth finding out the correlation between error correction and standard NLP tasks.

## 6.2 Methodology

To study the correlation between OCR post correction and NLP tasks, we derive the tags and named entities of OCR-ed files before feeding them to our seq2seq model. We repeat this procedure on the outputs of our model and compare them with tags and entities of the original OCR-ed text.

According to the tables of Chapter 3, we assume that the model notifies the position of errors; then, with the help of a human-assistant, the text will be corrected. We look into

such output for further analysis. We process each file separately and apply macro-level averaging to calculate the means of errors.

The stages of NLP tasks consists of sentence alignment, tokenization, part-of-speech tagging, and named entity recognition. In the sentence alignment stage, the detection of sentence boundary is crucial. We create pairs of aligned sentences per file. One pair consists of one sentence from OCR-ed text and the other from its corresponding GS/output of the model. Afterward, tokens are split by white space. Having a list of tokens per sentence of each pair, we use NLTK tool to identify POS tags. NLTK [16] is a Python NLP toolkit and is well- established in the research community. Part-of-Speech tagging assigns parts of speech to each word, such as noun, verb, adjective, etc. For each pair, we make a list of tags for OCR and GS. The label and order of tags must be identical. Otherwise, we detect it as one error. For each label from GS which there is no equivalent in OCR, we add up the number of errors.

Named Entity Recognition classifies named entities that are present in a text into predefined categories like individuals, companies, places, organization, cities, dates, product terminologies, etc. adding semantic knowledge to our content. The significance of such a task arises when we incline to categorize documents automatically in defined hierarchies and enable smooth content discovery. It also would be beneficial for automatic recommendation process. NLTK's named entity chunker is based on a supervised machine learning algorithm – Maximum Entropy Classifier. Its model is trained on ACE corpus with the exact entity types which we are interested in: "PERSON", "LOCATION" and "ORGANIZATION". Name Entity Recognition task exploits the outcomes of POS tagging component. 'NNP'(proper nouns, singular) and 'NNPS' (proper nouns, plural) tags are selected and grouped into predefined categories above. We check how effective our model is to correct proper nouns.

In addition to POS tagging and Named Entity Recognition, we also look into adjectives which are useful for Sentiment Analysis. Sentiment Analysis refers to the practice of applying Natural Language Processing and Text Analysis techniques to identify and extract subjective information from a piece of text. With Sentiment Analysis from a text analytics point of view, we are essentially concerning a topic in a piece of text and its polarity; whether it's positive, negative or neutral. Here, our focus is the number of adjectives detected before and after post-correction.

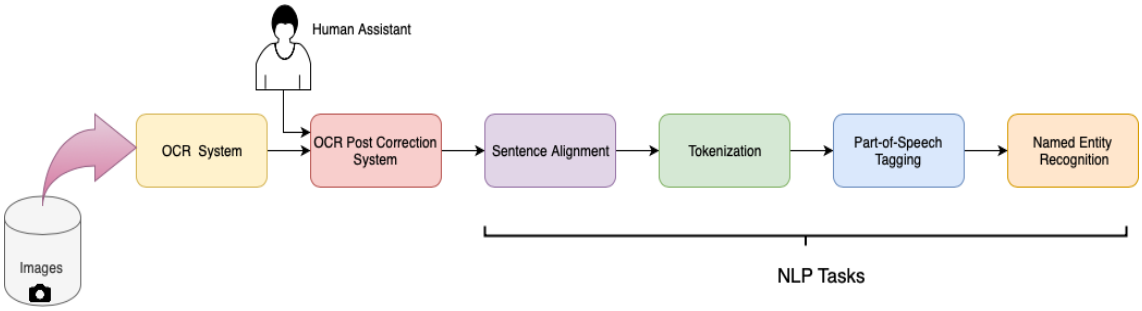



**Figure 6.1:** Text Analysis Pipeline

## 6.3 Experiments and Result Analysis

We picked English evaluation set of ICDAR 2017 to measure the impact of standard NLP tasks. It is essential to have almost a correct sentence to assign the accurate tag to each word. On the other hand, in order to correct one word, we are required to fix multi characters. Figure 6.2 shows the correlation between token level and character level correction, which is known as a delay effect. Therefore, one model may perform well in character level correction while it lacks satisfactory in word level and also standard NLP tasks.

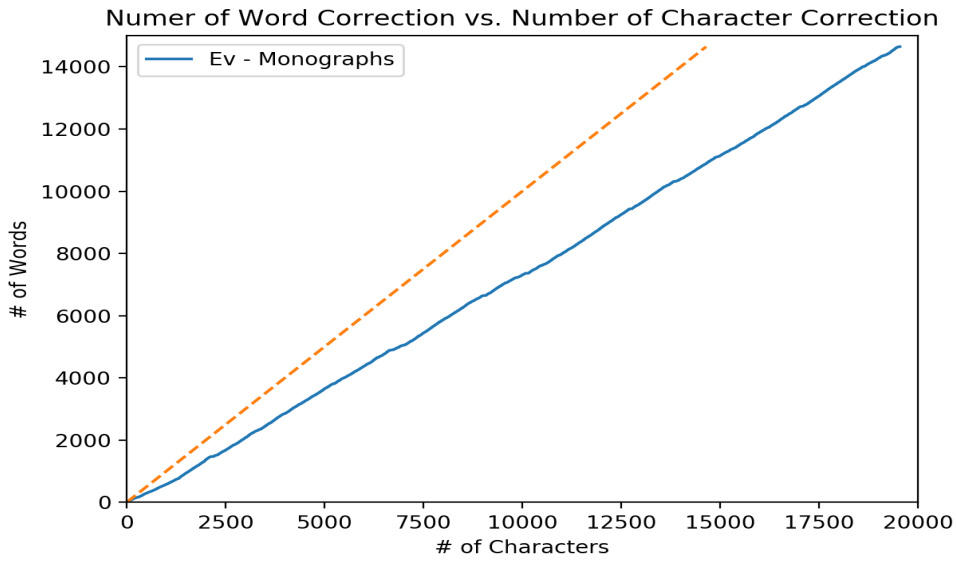



**Figure 6.2:** Number of Word Correction vs. Number of Character Correction

Figure 6.1 depicts the effect of OCR post-correction on POS tagging. The automatic

model flagged 92% of errors. Afterward, the assistant carried out the correction task. We assume that the human assistant is error-prone, and the error varies between 0 to 90%. In the ideal case, the assistant revises all flagged errors, which leads to 95% F1 as the final performance. We assessed 81 files individually and plotted the file improvement accordingly. The figure is given as a percentage of improvement vs. correction percentage. By improvement, we express what percentage of errors are corrected in accordance with the number of errors in original OCR-ed documents.

As we mentioned before, some systematic errors in ICDAR 2017 are related to punctuations making sentence boundary detection unreliable. In such cases, the errors will be cascaded from sentence alignment to the end, Table 6.1. The rationale behind error increase in files 33, 48, 61, 75, 76, 77, 78, 79, and 81 is a misalignment of sentences. We dropped those files in the evaluation of POS tagging since we decided to invest time on the POS tags rather than doing the alignment.

According to Figure 6.3, the pattern of document improvements is ascending with a dramatic increase in early stages and climbed more slowly to 80%. It is evident that even 10% correction (90% human error) result in 40% POS tag improvement in half of the files. Even with the model with 10% performance in correction, we can gain a significant amount. Implementing such a system is not unrealistic. On the other hand, for 80% improvement, we need almost 80% correction, which is too high.

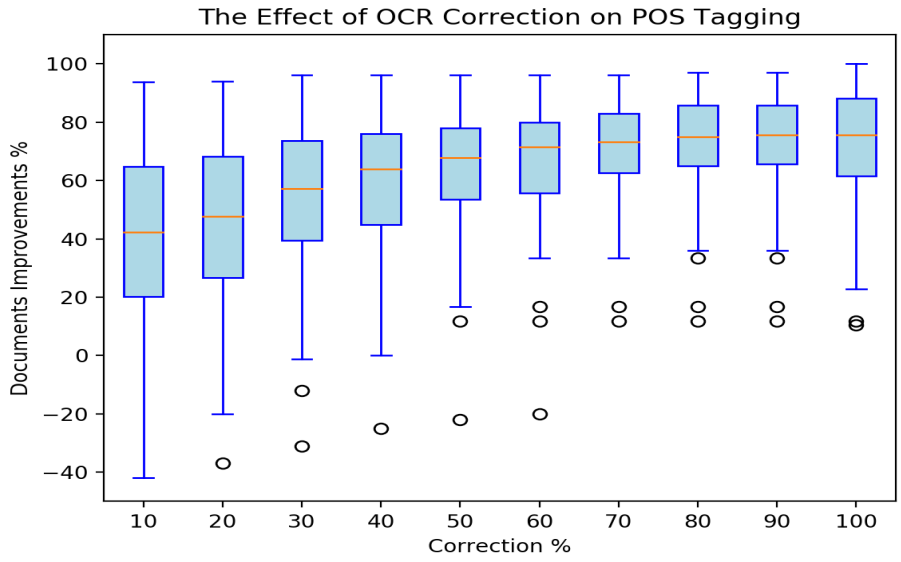



**Figure 6.3:** The Effect of OCR Correction on POS Tagging

Similarly, we draw the plot for Named Entity Recognition, Figure 6.4. We have data sparsity and less than 5% of errors related to named entities in most of the documents. The OCR system has embedded post-correction within the pipeline, which includes dictionaries. Since most of the noun errors were corrected there, now we encounter fewer noun errors. We can infer that with 10% correction, almost all named entities are corrected in the half of the documents. Moving from left to right in the x-axis, we only observe the interquartile range declines with quite a low rate.

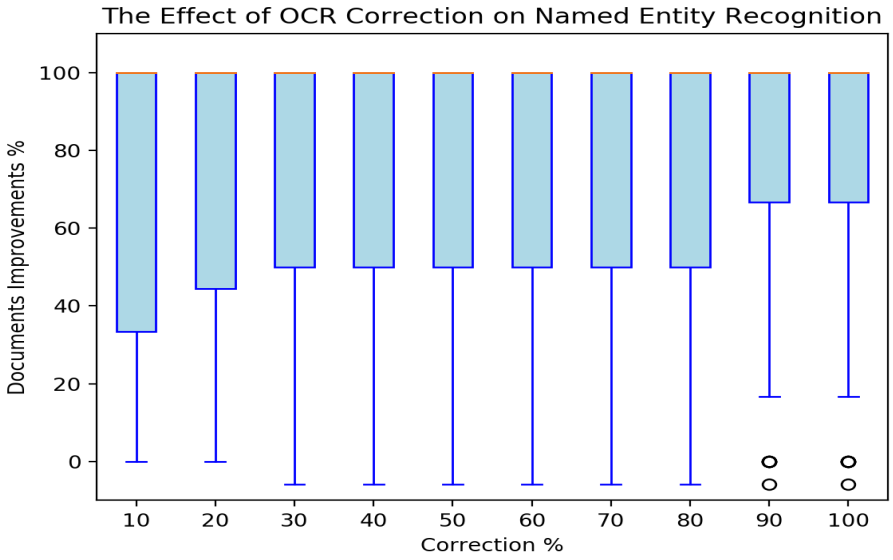



**Figure 6.4:** The Effect of OCR Correction on Named Entity Recognition

In the last examination, we looked into the improvement of documents from the percentage of adjective detection perspective. Although we didn't follow the approaches for sentiment analysis, we quantified the requirements for such a task, adjectives. Analogous to named entities, the number of errors that are adjectives are quite negligible (less than 1%) since they were fixed in the OCR system itself. According to Figure 6.5, we are able to correct nearly 80% of all adjectives with 10% correction. Furthermore, 30% correction gives us 100% for half of the files.

## 6.4 Conclusion

In this chapter, we addressed the last research question: *How do improvement in OCR quality impact downstream NLP tasks, such POS-tagging and NER?* We assess the worth

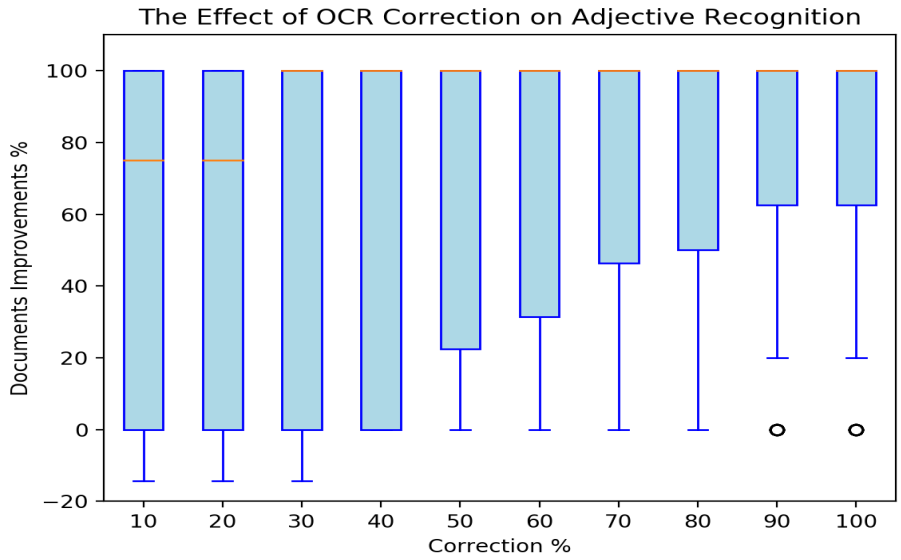



**Figure 6.5:** The Effect of OCR Correction on Adjective Recognition

of OCR post-correction by figuring out its effect on standard NLP tasks. We measured the errors in terms of POS tags, named entities, and adjectives. We can't generalize our findings of this chapter because the impact of NLP tasks is dependent on the performance of NLTK and the data in hand. Nonetheless, **firstly**, we could achieve a 40% improvement within half of the files in reducing the rate of tag errors by 10% correction. With the same correction rate, we also improved the documents named entities and adjective up to 100% and 80%, respectively. **Secondly**, we observed that we have data sparsity in terms of the number of the noun and adjective errors in the OCR-ed documents. **Thirdly**, we claimed that investment on the automatic OCR post-correction system is valuable even for a model with 10-30% performance. Implementing a model with such an achievement is feasible and brings great value.

**Table 6.1:** The Number of POS Tag Errors per File in Monographs: Before and After Correction

| Filename | POS Tags | # Errors | | Filename | POS Tags | # Errors | |
|---|---|---|---|---|---|---|---|
| | | Before | After | | | Before | After |
| 1.txt | 329 | 82 | 5 | 42.txt | 312 | 22 | 2 |
| 2.txt | 317 | 53 | 3 | 43.txt | 297 | 20 | 5 |
| 3.txt | 327 | 32 | 9 | 44.txt | 322 | 25 | 9 |
| 4.txt | 278 | 51 | 45 | 45.txt | 480 | 9 | 2 |
| 5.txt | 335 | 33 | 6 | 46.txt | 338 | 20 | 4 |
| 6.txt | 330 | 37 | 11 | 47.txt | 340 | 19 | 3 |
| 7.txt | 341 | 16 | 4 | 48.txt | 309 | 56 | 9 |
| 8.txt | 314 | 20 | 7 | 49.txt | 332 | 23 | 11 |
| 9.txt | 335 | 42 | 6 | 50.txt | 321 | 14 | 6 |
| 10.txt | 335 | 21 | 6 | 51.txt | 308 | 19 | 5 |
| 11.txt | 345 | 21 | 5 | 52.txt | 363 | 142 | 8 |
| 12.txt | - | - | - | 53.txt | 330 | 128 | 18 |
| 13.txt | 295 | 20 | 11 | 54.txt | 331 | 8 | 1 |
| 14.txt | 319 | 27 | 2 | 55.txt | 317 | 81 | 3 |
| 15.txt | 326 | 15 | 2 | 56.txt | 326 | 18 | 8 |
| 16.txt | 302 | 23 | 0 | 57.txt | 337 | 47 | 1 |
| 17.txt | 263 | 7 | 2 | 58.txt | 311 | 9 | 2 |
| 18.txt | 255 | 32 | 6 | 59.txt | 320 | 50 | 9 |
| 19.txt | 336 | 31 | 16 | 60.txt | 348 | 15 | 9 |
| 20.txt | 300 | 42 | 3 | 61.txt | 319 | 39 | 159 |
| 21.txt | 297 | 14 | 1 | 62.txt | 310 | 21 | 6 |
| 22.txt | 267 | 44 | 3 | 63.txt | 320 | 68 | 2 |
| 23.txt | 288 | 18 | 7 | 64.txt | 317 | 15 | 5 |
| 24.txt | 301 | 79 | 4 | 65.txt | 348 | 28 | 12 |
| 25.txt | 269 | 15 | 3 | 66.txt | 324 | 18 | 8 |
| 26.txt | 274 | 31 | 10 | 67.txt | 300 | 23 | 8 |
| 27.txt | 311 | 26 | 3 | 68.txt | 338 | 12 | 5 |
| 28.txt | 209 | 12 | 4 | 69.txt | 300 | 23 | 5 |
| 29.txt | 225 | 22 | 9 | 70.txt | 349 | 19 | 7 |
| 30.txt | 220 | 77 | 69 | 71.txt | 317 | 36 | 20 |
| 31.txt | 392 | 48 | 37 | 72.txt | 342 | 52 | 6 |
| 32.txt | 305 | 25 | 16 | 73.txt | 322 | 34 | 6 |
| 33.txt | 252 | 21 | 91 | 74.txt | 321 | 27 | 2 |
| 34.txt | 267 | 11 | 0 | 75.txt | 465 | 89 | 309 |
| 35.txt | 223 | 11 | 7 | 76.txt | 580 | 145 | 473 |
| 36.txt | 311 | 24 | 2 | 77.txt | 541 | 110 | 316 |
| 37.txt | 319 | 26 | 5 | 78.txt | 517 | 61 | 105 |
| 38.txt | 345 | 34 | 9 | 79.txt | 523 | 108 | 384 |
| 39.txt | 278 | 30 | 8 | 80.txt | 504 | 100 | 17 |
| 40.txt | 330 | 99 | 3 | 81.txt | 132,240 | 26,485 | 113,560 |
| 41.txt | 320 | 15 | 6 | | | | |

# Chapter 7

# Conclusion

In the current research project, we studied the following problem:

**Main Research Problem** *Can we improve Optical Character Recognition quality by post-correction?*

We made an effort to enhance the quality of OCR-ed documents by exploiting various methods and breaking it down into multiple sub-questions. We picked the dataset of ICDAR 2017 and ICDAR 2019 for our experiments.

Given two genres for our dataset, monographs, and periodicals, we derived token level and character level statistics of errors for the English section. Although the error rate in the periodicals exceeded monographs, the pattern of errors for both genres is almost identical. The majority of erroneous tokens are composed of 1-4 characters. Furthermore, the number of tokens with a single character error is higher than the remaining. Centering on characters, we observed errors related to character format, case insensitivity, punctuations, and segmentation issues with the majority of substitution cases. Such statistics motivated us to design a character level system for fixing the errors.

Designing the character level seq2seq model, we contributed more than 92% recall for the error detection task. This accomplishment made us interpose a human assistant in the OCR post-correction system to correct the flagged errors. We also reached the conclusion that training the model on each genre individually and also on the combination makes no difference in the outcome of the model. We concluded that even with a perfect model for estimating the quality of OCR, we need the help of an assistant to perform the correction task.

We stepped forward to assess the impact of transfer learning and fine-tuning on boosting the performance of the seq2seq model. Making use of English ICDAR 2017, we trained

various models for English ICDAR 2019. Transfer learning guarantees acceptable performance for the error detection task. Although the fine-tuning gains a better score than the baseline model, the model which was trained on ICDAR 2019, we must take the data used for fine-tuning into consideration. The closer the data into the pre-trained model, the greater the outcomes of the model would be. The comparison between transfer learning and fine-tuning demonstrated that we could utilize pre-trained models in different use cases without the need for additional training and money investment.

Due to the usefulness of transfer learning, we motivated to extend our seq2seq model across languages. Instead of training a fragile model per language due to the lack of data for some languages, we took advantage of the entire data as a whole. Character level model provided the opportunity for us to do such an experiment since our solution is language independent. We only evaluated our model for French and English. Transfer learning incurs better precision and less recall. Moreover, the outcomes of our experiments proved the practicability of fine-tuning in comparison with training in a specific language. We also recognized that OCR errors are independent of a particular languages.

We investigated the practicality of the OCR post-correction system by measuring its impact on standard NLP tasks such as POS tagging and Named Entity Recognition. We figured out even 10% correction leads into 40% improvement in revising POS tags and near to ideal improvement in Named Entity Recognition within half of the data files. Furthermore, the number of errors which are adjectives or nouns is infrequent. Nonetheless, 10% correction scored almost 80% improvement in fixing the adjectives in half of the files. Having a fully automated system with 10-30% -which is feasible- could bring excellent value for NLP tasks.

Within the current thesis, we contributed a novel approach for estimating the quality of OCR-ed documents with high recall. However, the system is semi-automated taking advantage of an assistant for the correction task. We enriched our solution with multi-lingual training data and methods such as transfer learning and fine-tuning to gain better results with less effort. Although including a human in the loop seems unavoidable, even an autonomous post-correction system with a tiny OCR improvement can result in considerable growth in NLP tasks.

There are many options we can think about as future work. We focused on ICDAR 2017 and English ICDAR 2019 while we are able to work on other datasets as well as different languages of ICDAR 2019. During the current project, we only focused on seq2seq model. However, it is worthy of inserting dictionaries and n-gram statistics to increment the accuracy of OCR-ed documents. To validate the outcomes of different experiments

for ICDAR 2019 competition, we should double check the measurements after the release of the testing set. We can also go over two languages and repeat the fine-tuning step up to the point that model learn all 10 European languages of ICDAR 2019. In the NLP pipeline, we assessed the model for simple NLP tasks. We are able to study the models in a more complex pipeline like question answering, topic detection, etc.

# References


[1] H. Afli, L. Barrault, and H. Schwenk. Ocr error correction using statistical machine translation. *Int. J. Comput. Linguistics Appl.*, 7:175–191, 2015. 2, 10

[2] H. Afli, Z. Qiu, A. Way, and P. Sheridan. Using smt for ocr error correction of historical texts. In *LREC*, 2016. 6

[3] H. Afli, Z. Qiu, A. Way, and P. Sheridan. Using SMT for OCR error correction of historical texts. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 962–966, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA). URL `https://www.aclweb.org/anthology/L16-1153`. 4, 10

[4] C. Amrhein and S. Clematide. Supervised ocr error detection and correction using statistical and neural machine translation methods. *Journal for Language Technology and Computational Linguistics (JLCL)*, 33(1):49–76, 2018. ISSN 0175-1336. URL `https://doi.org/10.5167/uzh-162394`. 5

[5] E. Atwell and S. Elliott. Dealing with ill-formed english text. *Comput. Anal. Engl.: Corpus-Based App.*, 01 1987. 4

[6] S. Bukhari, A. Kadi, M. Jouneh, F. Mir, and A. Dengel. anyocr: An open-source ocr system for historical archives. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, pages 305–310, Los Alamitos, CA, USA, nov 2017. IEEE Computer Society. doi: 10.1109/ICDAR.2017.58. URL `https://doi.ieeecomputersociety.org/10.1109/ICDAR.2017.58`. 1

[7] G. Chiron, A. Doucet, M. Coustaty, and J. Moreux. Icdar2017 competition on post-ocr text correction. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 01, pages 1423–1428, Nov 2017. doi: 10.1109/ICDAR.2017.232. 4, 5, 6

[8] G. Chiron, A. Doucet, M. Coustaty, M. Visani, and J.-P. Moreux. Impact of ocr errors on the use of digital libraries: Towards a better access to information. In *Proceedings of the 17th ACM/IEEE Joint Conference on Digital Libraries*, JCDL '17, pages 249–252, Piscataway, NJ, USA, 2017. IEEE Press. ISBN 978-1-5386-3861-3. URL http://dl.acm.org/citation.cfm?id=3200334.3200364. 10

[9] J. Chung, Çaglar Gülçehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *ArXiv*, abs/1412.3555, 2014. v, 23

[10] E. D'hondt, C. Grouin, and B. Grau. Generating a training corpus for ocr post-correction using encoder-decoder model. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1006–1014. Asian Federation of Natural Language Processing, 2017. URL http://aclweb.org/anthology/I17-1101. 1, 6

[11] R. Dong and D. Smith. Multi-input attention for unsupervised ocr correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2363–2372. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/P18-1220. 6

[12] J. Evershed and K. Fitch. Correcting noisy ocr: context beats confusion. In *DATeCH*, 2014. 5

[13] J. Evershed and K. Fitch. Correcting noisy ocr: Context beats confusion. *ACM International Conference Proceeding Series*, 05 2014. doi: 10.1145/2595188.2595200. 1

[14] M. Généreux and D. Spano. Nlp challenges in dealing with ocr-ed documents of derogated quality. 07 2015. 2

[15] C. Hokamp. Ensembling factored neural machine translation models for automatic post-editing and quality estimation. In *WMT*, 2017. 6

[16] R. Jiang, R. E. Banchs, and H. Li. Evaluating and combining name entity recognition systems. In *Proceedings of the Sixth Named Entity Workshop*, pages 21–27, Berlin, Germany, Aug. 2016. Association for Computational Linguistics. doi: 10.18653/v1/W16-2703. URL https://www.aclweb.org/anthology/W16-2703. 50

[17] I. Kissos and N. Dershowitz. Ocr error correction using character correction and feature-based word classification. In *2016 12th IAPR Workshop on Document Analysis Systems (DAS)*, pages 198–203, April 2016. doi: 10.1109/DAS.2016.44. 1, 4, 5, 10

[18] P. Koehn. Neural machine translation. *ArXiv*, abs/1709.07809, 2017. 4

[19] O. Kolak and P. Resnik. Ocr error correction using a noisy channel model. In *Proceedings of the Second International Conference on Human Language Technology Research*, HLT '02, pages 257–262, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc. URL `http://dl.acm.org/citation.cfm?id=1289189.1289208`. 5

[20] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521:436–44, 05 2015. doi: 10.1038/nature14539. v, 23

[21] D. P. Lopresti. Optical character recognition errors and their effects on natural language processing. *International Journal on Document Analysis and Recognition (IJDAR)*, 12:141–151, 2009. 49

[22] T.-T.-H. Nguyen, M. Coustaty, A. Doucet, A. Jatowt, and N.-V. Nguyen. Adaptive edit-distance and regression approach for post-ocr text correction. In M. Dobreva, A. Hinze, and M. Žumer, editors, *Maturity and Innovation in Digital Libraries*, pages 278–289, Cham, 2018. Springer International Publishing. ISBN 978-3-030-04257-8. 1, 6

[23] M. Reynaert. Ocr post-correction evaluation of early dutch books online - revisited. In Calzolari, editor, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 967–974. ELRA, 2016. 4

[24] S. Schulz and J. Kuhn. Multi-modular domain-tailored ocr post-correction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2716–2726. Association for Computational Linguistics, 2017. doi: 10.18653/v1/D17-1288. URL `http://aclweb.org/anthology/D17-1288`. 5, 6

[25] S. Schuster, S. Gupta, R. Shah, and M. Lewis. Cross-lingual transfer learning for multilingual task oriented dialog. In *NAACL-HLT*, 2018. 2

[26] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages

3104–3112. Curran Associates, Inc., 2014. URL http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf. 5

[27] X. Tong and D. A Evans. A statistical approach to automatic ocr error correction in context. *Proceedings of the Fourth Workshop on Very Large Corpora*, 01 1996. 5

[28] M. Volk, L. Furrer, and R. Sennrich. Strategies for reducing and correcting ocr errors. In C. Sporleder, A. van den Bosch, and K. Zervanou, editors, *Language Technology for Cultural Heritage*, pages 3–22, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. 4

[29] T. Young, D. Hazarika, S. Poria, and E. Cambria. Recent trends in deep learning based natural language processing [review article]. *IEEE Computational Intelligence Magazine*, 13:55–75, 2018. 20

[30] T. Young, D. Hazarika, S. Poria, and E. Cambria. Recent trends in deep learning based natural language processing [review article]. *IEEE Computational Intelligence Magazine*, 13:55–75, 2018. 23