OCR Error Correction:
State-of-the-art vs An NMT Based Approach

Kareem Mokhtar*, Syed Saqib Bukhari*, Andreas Dengel
German Research Center for Artificial Intelligence (DFKI), Germany
University of Kaiserslautern, Germany
{saqib.bukhari, andreas.dengel}@dfki.de

Abstract—Although the performance of the state-of-the-art OCR systems is very high, they can still introduce errors due to various reasons. And when it comes to historical documents with old manuscripts the performance of such systems gets even worse. That is why Post-OCR error correction has been an open problem for many years. Many state-of-the-art approaches have been introduced thorough the recent years.

This paper contributes to the field of Post-OCR Error Correction by introducing two Novel deep learning approaches to improve the accuracy of OCR systems, and a post processing technique that can further enhance the quality of the output results. These approaches are based on Neural Machine Translation and were motivated by the great success that deep learning introduced to the field of Natural Language Processing. Finally, we will compare the state-of-the-art approaches in Post-OCR Error Correction with the newly introduced systems and discuss the results.

I. INTRODUCTION

Since the invention of the first machine until today, the main goal for humanity was to replicate the human functions and produce machines that are capable of performing tasks with high speed and an accuracy near or even better than humans. One of these abilities is reading. In the recent five decades Machine Reading has evolved significantly, and the need for such technology increased as well. Today Optical character recognition (OCR) systems are capable of converting images of typed, handwritten or printed text into machine-encoded searchable text, whether from a scanned document, a photo of a document, a scene-photo or from subtitle text superimposed on an image. Despite the huge success of OCR systems they are still no match for humans as they produce errors while reading documents. Mainly these errors are produced due to various reasons like poor image quality or the complex structure of a page Figure 1.a. Moreover, these errors becomes more evident while dealing with historical documents as such documents can contain some challenging characteristics and some degradations in quality like historical fonts including ligatures, historical spelling variants, somewhat displaced characters (resulting from historical printing processes), fuzzy character boundaries due to ink creep into the paper over time, paper degradation resulting in dark backgrounds, blotches, cracks, dirt, and bleed through from the following page Figure 1.b. A satisfactory OCR error rate while scanning big documents like books for example can contribute in lowering the time and cost of transcribing such documents manually. Moreover such OCR systems can be reliable in applications like the digitization of historical documents to preserve its content. Therefore comes the need of reducing the errors in the OCR output as much as possible.

In this paper, First of all, we are going to introduce two novel deep learning architectures that is used for Post-OCR correction. Moreover, we will introduce a Post processing technique to help further improve the accuracy of the two models. Finally, these novel approaches are compared to the state-of-the-art techniques and discuss the results. The motivation behind choosing Deep Neural Networks to solve the problem of Post-OCR Error Correction is mainly the success that was achieved by them in different fields over the last decade and the incredible results that was in the field of Natural Language processing in the recent years. The second reason is that the majority of research papers about OCR Correction are using the state-of-the-art techniques and mainly Statistical Language Modeling. The rest of the paper is organized as follows. Section II presents the brief overview of Neural Machine Translation (NMT). Our OCR correction methodology is presented in Section III. Section IV presents experiments and results which followed by conclusion in Section V.

*These two authors contributed equally.
Neural Machine Translation (NMT) is a machine translation approach that is using Neural Networks. For many decades Machine translation was done mainly by dividing the given sentence into multiple parts and then try to provide a translation for each one. These approaches was causing the translation to be lacking the fluency of the language and most of the time the translation is out of order in the target language due to the different grammar. The recent developments in Statistical Machine Translation lead to more natural translation by finding new approaches in reordering its output sentences to abide the grammar rules. However, all of these approaches was far from the natural behavior of humans which is reading the whole source sentence understand the meaning and then construct the target sentence based on the meaning only ignoring the one-to-one translations of words. That is exactly how Neural machine translation works.

Neural Machine Translation works in the following flow. First The source sentence is fed to the network’s encoder that outputs an array of numerical values. These values represents meaning of the sentence independently of the used dictionary. After that, these values are fed into the network’s decoder that can construct a new sentence with the same meaning however with the vocabulary and grammar of the target language. This flow is called encoder-decoder architecture Figure 2.

These encoders and decoders can be implemented using different neural network architectures. However, because we are dealing with sequential data (sentences) it is only natural to work with Recurrent Neural Networks (RNN) for the encoder and decoder. Recurrent Neural Networks have many architectural flavors in terms of the RNN cells used either Long-Short-term Memory (LSTM) [1] or a gated recurrent unit (GRU) [2], depth of the network and more.

The RNN cell that is used in the proposed model is a Long-Short-term Memory (LSTM). This choice is motivated by the outstanding performance of the LSTM recurrent neural networks in the fields of speech recognition, language modeling, translation and image captioning.

In this paper, OpenNMT [3] will be used to build the neural network architecture. OpenNMT is an open source toolkit for Neural Machine Translation, that gives the user the ability to customize the RNN to suite the application it is used for.

III. METHODOLOGY

In this section two different recurrent neural network architectures are presented to improve the accuracy of the OCR output. Those two architectures are almost the same however with a slight change in the input and output formats of the network. After that, we are going to introduce a post-processing approach that can help increase the accuracy of the introduced models.

A. Word Based Sequence-to-Sequence model

This model is based on the encoder-decoder recurrent neural network architecture. The idea is to treat the Post-OCR correction problem as a translation problem. This is done by considering the source language as the erroneous OCR output and the corrected output as the target language. This model works on a word level as it encodes the sentence as a sequence of words. The input dictionary is the set of words used in the OCR output. While the dictionary of the output is the set of words that are extracted during the preprocessing step from the target language.

In this model we will use an RNN with the following specifications:

- depth (Number of layers of LSTM cells inside encoder/decoder): 3
- Size of LSTM hidden states: 1024
- Word embedding size: 1024
- dropout percentage: 30%
- optimizer: Stochastic gradient descent
- learning rate: 1
- learning rate decay rate: 50%
- start learning decay at epoch: 8
- number of epochs: 20

The input for this model is a parallel corpus of the OCR output and the corresponding ground truth. Before feeding the input to the model to train first we have to pre-process it as follows:

- Each sentence should be tokenized so that every word is separate from any punctuation Figure 4.
- The dictionary of the source and the target languages are built (Set of words in each language).

This RNN architecture (Figure 3) have some advantages as well as some drawbacks. The following table I is showing some of them:
Don’t forget, He said "I will be there in 5 minutes" before leaving.

He is here.

Don’t forget, He said "I will be there in 5 minutes" before leaving.

Fig. 4. Word Based model input example

He <SPACE> is <SPACE> here.

Fig. 6. Character Based model input example

TABLE I
SHOWING THE ADVANTAGES AND THE DISADVANTAGES OF THE WORD BASED MODEL.

<table>
<thead>
<tr>
<th></th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>It learns the relations between the words that define a correct sentence</td>
<td>It doesn’t correct a word if it is not in the output dictionary</td>
</tr>
<tr>
<td></td>
<td>It learns the common word mistakes and their corrections</td>
<td>It doesn’t learn the mistakes on a character level</td>
</tr>
<tr>
<td></td>
<td>The corrected sequence doesn’t have to be the same length as the erroneous one</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The dictionary of the source and the target languages are built (Set of words in each characters).</td>
</tr>
</tbody>
</table>

This Character Based model (Figure 5) have some advantages as well as some drawbacks. The following table II is showing some of them:

### C. Word Level Normalization of predictions

Although the previous models on their own showed good output results Figure 7, it also sometimes when the input word is distorted to the limit that it is not identifiable the prediction output will be a totally different word with respect to the OCR input Figure 8. For that reason the Normalization concept was introduced.

The Normalization approach is a technique that is applied to the OCR erroneous output and the corresponding correction suggestion and it applies the following steps:

1) Align the OCR output with the neural network suggestion using Levenshtein edit distance, so that every word in the OCR output is processed with its corresponding word in the corrected output from the neural network.

2) loop over the words one by one and pick either the OCR output or the suggested correction based on the edit distance between both words as we set a dynamic threshold based on the word length. See the code 1

3) Output the resultant sentence.

Normalization can help solve the problem that appeared in Figure 8, and give the output that is in Figure 9

INPUT: dass es eine ganze Fenge Übereinstimmung zwischen

PRED : dass es eine ganze Menge Übereinstimmung zwischen

GOLD : dass es eine ganze Menge Übereinstimmung zwischen

Fig. 7. NMT good correction Example without normalization

INPUT: urid sich diegin gern beteiligen ’ ’ , sagte Lafontainedem

PRED : wird sich studig gern beteiligen ’ ’ , sagte Entwicklungspolitiker

GOLD : wird sich die Linke gern beteiligen ’ ’ , sagte Lafontaine dem

Fig. 8. NMT distorted correction Example without normalization
INPUT: würd sich dieinge gern beteiligen ’ ’, sagte Lafortainedem
PRED: würd sich ständig gern beteiligen ’ ’, sagte Entwicklungspolitiker
NORM: würd sich dieinge gern beteiligen ’ ’, sagte Lafortainedem
GOLD: würd sich die Linke gern beteiligen ’ ’, sagte Lafortaine dem

Fig. 9. NMT distorted correction Example with normalization

The code in Listing 1 is a JavaScript implementation of the normalization technique

```javascript
/* the variables input and pred contains the OCR output sentence and the predicted sentence respectively */
/* res will contain the result sentence words after normalization */
/* align is a function that takes two sentences and aligns them together using edit distance */
var alignment = align(input, pred);
input = alignment[0].split(' /s+/);
pred = alignment[1].split(' /s+/);
/* now we have two arrays of words that are aligned together */
for (var j = 0; j < input.length; ++j) {
  var edits = editDistance(input[j], pred[j]);
  var match = pred[j].length - edits;
  var threshold = 0;
  switch (input[j].length) {
    case 1:
    case 2:
    case 3: threshold = 1; break;
    case 4:
    case 5:
    case 6: threshold = 3; break;
    default: threshold = 5; break;
  }
  if (match >= threshold) {
    res.push(pred[j]);
  } else {
    res.push(input[j]);
  }
}
res = res.join(' ');
```

Listing 1. Normalization picking criteria

IV. EXPERIMENT DESIGN

In this section, we present the datasets that are used to test all of the approaches and how are they collected. After that, we explain the evaluation metrics that is used to evaluate the systems.

A. Datasets

We experimented with different datasets of different languages. In fact we used three datasets one dataset for each of the following languages English, German and Medieval Latin (15th century).

1) English dataset: This is a dataset of contemporary English. It consists of two parallel parts the OCR output and the corresponding ground truth.

This data is collected from two Main sources:

- UNLV dataset: The dataset contains 2889 pages of scanned document images from variety of sources (Magazines, News papers, Business Letter, Annual Report, ...etc). The scanned images are provided at 200 and 300 DPI resolution in bitonal, grey and fax format. There is ground truth data provided alongside the original dataset which contains manually marked zones; zone types are provided in text format.

- UW-3 dataset: The dataset consists of 1600 skew-corrected English document images with manually edited ground-truth of entity bounding boxes. These bounding boxes enclose page frame, text and non-text zones, text lines, and words. The type of each zone (text, math, table, figure, ...etc) is also marked.

The data was extracted from both datasets line by line by cropping each line on its own and then run anyOCR [4] on it. The motivation behind this cropping technique is to eliminate as much error as possible by forcing the reading order. finally, the data were combined into two parallel documents (OCR output and the ground truth) with a sentence in each line.

2) German dataset: This is a dataset of contemporary German it was collected from German newspapers using web scrapers. After that, these collected sentences were rendered one by one using fonts that mimics the effect of scanned documents (Figure 10). These sentences are fed to anyOCR and the output is combined into one file with a sentence in every line while appending the corresponding ground truth to the ground truth file.

3) Medieval Latin dataset: This is a Latin dataset that is collected from a Latin medieval book from the 15th century (Figure 11). The text in this book is written in an old font that uses lots of ligatures and old character forms.

This data was extracted by running anyOCR on these documents, and after getting the output manual alignment of the OCR sentences with the ground truth was done. So, we ended up with two files of parallel erroneous text with its corresponding ground truth.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>#Sentences</th>
<th>Training %</th>
<th>Validation %</th>
<th>Test %</th>
</tr>
</thead>
<tbody>
<tr>
<td>English</td>
<td>116951</td>
<td>90% (105257)</td>
<td>5% (5847)</td>
<td>5% (5847)</td>
</tr>
<tr>
<td>German</td>
<td>21076</td>
<td>70% (14754)</td>
<td>15% (3161)</td>
<td>15% (3161)</td>
</tr>
<tr>
<td>Latin</td>
<td>3175</td>
<td>80% (2541)</td>
<td>10% (317)</td>
<td>10% (317)</td>
</tr>
</tbody>
</table>

TABLE III
SHOWING STATISTICS ABOUT THE DATASETS

Weihnachten oder Hochzeiton etwas radikalste Abkehr von der damals
B. Evaluation Metrics

The process of evaluating how the suggested correction improved or deteriorated the accuracy of the OCR output is one of the key essentials to compare the newly introduced approach with the state-of-the-art approaches. One of the most used measurements in majority of the research papers in the field of OCR correction is the character error rate (CER) and the word error rate (WER) before and after applying the correction model on the OCR output.

Error rate is a metric that measures how different a given text from its ground truth in terms of the total number of Levenshtein edit operations (Insertions, Deletions and Substitutions) that are required to transform the text to its ground truth. This is done on a character/word level according to the following formula:

\[
CER/\text{WER} = \frac{I + D + S}{N} \times 100
\]  

(1)

The components of the equation are:

- **I**: The minimum number of character/word insertion operations required by the Levenshtein edit distance between the text and its ground truth.
- **D**: The minimum number of character/word deletion operations required by the Levenshtein edit distance between the text and its ground truth.
- **S**: The minimum number of character/word substitution operations required by the Levenshtein edit distance between the text and its ground truth.
- **N**: The total number of characters/words in the ground truth.

V. PERFORMANCE EVALUATION

In this section we present the experiments that was conducted and their results. We conducted experiments with six different approaches to get to compare the results of the newly introduced approaches with the state-of-the-art techniques. The used approaches are as follows:

- state-of-the-art approaches
  - Uni-gram language modeling [5] (Uni-gram)
- introduced novel approaches
  - Word based model (WBM)
  - Character based model (CBM)
  - Word based model with Normalization (CBM++)
  - Character based model with Normalization (CBM++)

The results of the experiments showed that the best performing model is the Character based model with normalization (CBM++) as it doesn’t need big training data to give good performance and also it can correct newly seen words by correcting individual characters.

### TABLE IV
RESULTS OF EXPERIMENTING WITH UNI-GRAM

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Base Error</th>
<th>Uni-gram</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CER</td>
<td>WER</td>
</tr>
<tr>
<td>English</td>
<td>8.71%</td>
<td>33.11%</td>
</tr>
<tr>
<td>German</td>
<td>10.65%</td>
<td>47.12%</td>
</tr>
<tr>
<td>Latin</td>
<td>2.87%</td>
<td>8.87%</td>
</tr>
</tbody>
</table>

### TABLE V
RESULTS OF EXPERIMENTING WITH SMT

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Base Error</th>
<th>SMT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CER</td>
<td>WER</td>
</tr>
<tr>
<td>English</td>
<td>8.71%</td>
<td>33.11%</td>
</tr>
<tr>
<td>German</td>
<td>10.65%</td>
<td>47.12%</td>
</tr>
<tr>
<td>Latin</td>
<td>2.87%</td>
<td>8.87%</td>
</tr>
</tbody>
</table>

### TABLE VI
RESULTS OF EXPERIMENTING WITH WBM

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Base Error</th>
<th>WBM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CER</td>
<td>WER</td>
</tr>
<tr>
<td>English</td>
<td>8.71%</td>
<td>33.11%</td>
</tr>
<tr>
<td>German</td>
<td>10.65%</td>
<td>47.12%</td>
</tr>
<tr>
<td>Latin</td>
<td>2.87%</td>
<td>8.87%</td>
</tr>
</tbody>
</table>

### TABLE VII
RESULTS OF EXPERIMENTING WITH WBM++

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Base Error</th>
<th>WBM++</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CER</td>
<td>WER</td>
</tr>
<tr>
<td>English</td>
<td>8.71%</td>
<td>33.11%</td>
</tr>
<tr>
<td>German</td>
<td>10.65%</td>
<td>47.12%</td>
</tr>
<tr>
<td>Latin</td>
<td>2.87%</td>
<td>8.87%</td>
</tr>
</tbody>
</table>
TABLE VIII
RESULTS OF EXPERIMENTING WITH CBM

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Base Error</th>
<th>CBM</th>
<th>Relative Improvement (CER)</th>
<th>Relative Improvement (WER)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CER</td>
<td>WER</td>
<td>CER</td>
<td>WER</td>
</tr>
<tr>
<td>English</td>
<td>8.71%</td>
<td>33.11%</td>
<td>7.63%</td>
<td>22.34%</td>
</tr>
<tr>
<td>German</td>
<td>10.65%</td>
<td>47.12%</td>
<td>15.12%</td>
<td>34.01%</td>
</tr>
<tr>
<td>Latin</td>
<td>2.87%</td>
<td>8.87%</td>
<td>3.33%</td>
<td>8.82%</td>
</tr>
</tbody>
</table>

TABLE IX
RESULTS OF EXPERIMENTING WITH CBM++

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Base Error</th>
<th>CBM++</th>
<th>Relative Improvement (CER)</th>
<th>Relative Improvement (WER)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CER</td>
<td>WER</td>
<td>CER</td>
<td>WER</td>
</tr>
<tr>
<td>English</td>
<td>8.71%</td>
<td>33.11%</td>
<td>6.23%</td>
<td>23.15%</td>
</tr>
<tr>
<td>German</td>
<td>10.65%</td>
<td>47.12%</td>
<td>8.87%</td>
<td>33.69%</td>
</tr>
<tr>
<td>Latin</td>
<td>2.87%</td>
<td>8.87%</td>
<td>2.93%</td>
<td>9.18%</td>
</tr>
</tbody>
</table>

Fig. 12. Comparison between different models Accuracy on a character level.

Fig. 13. Comparison between different models Accuracy on a word level.

VI. CONCLUSION

In this paper we experimented with the state-of-the-art approaches in OCR error correction. After that, based on the success of LSTM Recurrent Neural Networks in the field of Natural language processing, we introduced two new deep neural networks that can solve the same problem. The first model we introduced was the word based neural machine translation model. This word based model had the potential to outperform the rest of approaches, however due to the small data used in the training phase, the results were significantly bad. On the other hand, the second model we introduced was the character based neural machine translation model model showed significant improvement compared to the state-of-the-art approaches over all the used datasets even though the training data is considered relatively small. Moreover, we applied a normalization technique as a post processing step to enhance the accuracy of our deep learning models and as a result the accuracy of the models improved to a point where it outperformed the state-of-the-art approaches significantly. That makes it evident that the approaches we introduced are preforming better than the state-of-the-art approaches and can be used to improve the accuracy of OCR systems.

REFERENCES


