2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)

Feedback Learning: Automating the Process of Correcting and Completing the Extracted Information

1st Khurram Azeem Hashmi
†German Research Center for Artificial Intelligence (DFKI)
Kaiserslautern, Germany
khurram_azeem.hashmi@dfki.de

2nd Rakshith Bymana Ponnappa
†German Research Center for Artificial Intelligence (DFKI)
Kaiserslautern, Germany
rakshith.bymana_ponnappa@dfki.de

3rd Syed Saqib Bukhari
†German Research Center for Artificial Intelligence (DFKI)
Kaiserslautern, Germany
saqib.bukhari@dfki.de

4th Martin Jenckel
†German Research Center for Artificial Intelligence (DFKI)
Kaiserslautern, Germany
martin.jenckel@dfki.de

5th Andreas Dengel
†German Research Center for Artificial Intelligence (DFKI)
Kaiserslautern, Germany
andreas.dengel@dfki.de

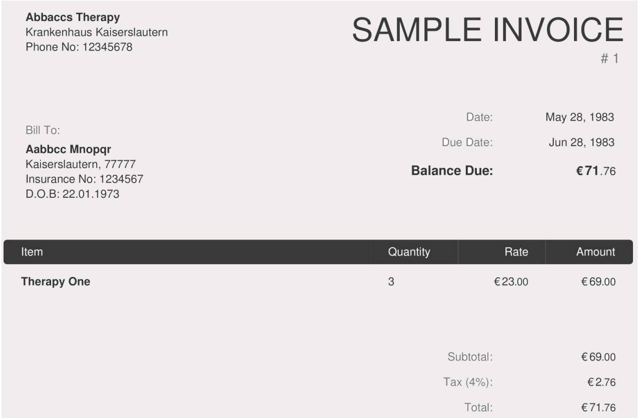
***Abstract*—In recent years, with the increasing usage of digital media and advancements in deep learning architectures, most of the paper-based documents have been revolutionized into digital versions. These advancements have helped state-of-the-art information extraction and digital mailroom technologies become progressively efficient. Even though many efficient post-Information Extraction (IE) error rectification methods have been introduced in the recent past to improve the quality of digitized documents. They are still imperfect and they demand improvements in the area of context-based error correction, specifically when we are dealing with the documents involving sensitive information such as invoices. This paper describes a self-correction approach based on the sequence to sequence Neural Machine Translation (NMT) as applied to rectify the incorrectness in the results of any information extraction approach such as Optical Character Recognition (OCR). We accomplished this approach by exploiting the concepts of sequence learning with the help of feedback provided during each cycle of training. Finally, we have compared state-of-the-art post-OCR error correction methods with our feedback learning approach. Our empirical results have outperformed state-of-the-art post-OCR error correction methods.**

***Keywords*-Document Understanding, Post IE Error Correction and Completeness, Sequence to Sequence Neural Machine Translation**

I. Introduction

From the early stages of computers to the current digital era, computers have been remarkably transformed and upgraded. Among many applications, the main purpose of the computer is to process information. In this age of automation, where the primary aim is to digitize everything, one of the essential tasks is to capture and process all kinds of documents. While the documents containing texts are rather interpretable, working on the scanned images of the documents is not an effortless operation.

An example of a sample invoice can be seen in Figure 1. If we want to extract the following information such as first name, last name, date of birth, insurance number and hospital name from the image, some of the possible errors in the extracted information along with the respective ground truth are shown in Figure 2.



Abbaccs Therapy
Krankenhaus Kaiserslautern
Phone No: 12345678

SAMPLE INVOICE

1

Bill To:

Aabbcc Mnopqr
Kaiserslautern, 77777
Insurance No: 1234567
D.O.B: 22.01.1973

Date:	May 28, 1983
Due Date:	Jun 28, 1983
Balance Due:	**€71**.76

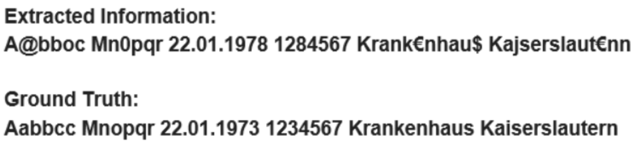
Item	Quantity	Rate	Amount
Therapy One	3	€23.00	€69.00

Subtotal:	€69.00
Tax (4%):	€2.76
Total:	€71.76

Figure 1: Synthetic sample invoice image containing information of the patient.

These erroneous words can affect the processing of data specifically when the data is important such as personal

978-1-7281-5054-3/19/$31.00 ©2019 IEEE
DOI 10.1109/ICDARW.2019.40091



Extracted Information:
A@bboc Mn0pqr 22.01.1978 1284567 Krank€nhau$ Kajserslaut€nn

Ground Truth:
Aabbcc Mnopqr 22.01.1973 1234567 Krankenhaus Kaiserslautern

Figure 2: Extracted information from the synthetic sample invoice (as shown in Figure 1) and it's corresponding ground truth. Some of the errors in the extracted information can be observed here.

information of clients for a company. They also lead to manual labor where a person has to read all the extracted information, distinguish the errors and correct the mistakes every time. Our approach stands to reduce the human effort involved in correcting the errors and completing the missing information from the extracted data.

Plenty of research has been conducted in the area of error corrections with various techniques. Some of those techniques involve machine learning algorithms to rectify the textual errors obtained from OCR [10],[11],[9]. In another approach, they propose a method to generalize OCR error correction using ensembling methods [12]. It has also been illustrated that the errors generated from the IE systems are more diverse than the handwriting errors [5], [8]. Post-OCR errors have also been corrected using Google's online spelling suggestion [17] and Simulated Annealing (SA) [6]. In one of the previous works [13], NMT has already been used in correcting post-OCR errors in the historical documents but our approach differs by combining NMT with a feedback learning technique.

In this paper, the data consists of customer and company profiles based on the type of invoice processed. For example, in a health care invoice, the data might have all the personal information such as *First name*, *Last name*, *Date of birth*, *Hospital Address*, *Type of Medicine* and so on. One of the biggest problem in these use cases is that we cannot rely on pre-trained language models because most of the values in the data are proper nouns. To overcome this problem we have introduced a new technique called *Feedback Learning* over sequence to sequence learning technique.

The rest of the paper is organized as follows. Section II explains the working of NMT and Section III describes the Feedback Learning process. Section IV defines the methodology used in detail. Section V illustrates the design and experiments. In Section VI we discuss the evaluation of the obtained results along with comparison with state of the art approaches. Section VII concludes the paper.

II. Sequence to Sequence Neural Machine Translation

Neural Machine Translation (NMT) is a machine translation method that uses deep neural networks. NMT is based on the sequence-to-sequence models [16], [2]. The sequence-to-sequence models mimic how human interprets any sentence. Humans read the entire sentence, interpret the meaning, and then map those words into respective translation.

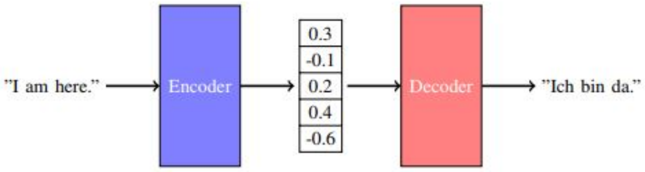


Figure 3: A simple representation of an encoder-decoder architecture [13] which translates the English sentence "I am here" into the German language.

The implementation of the decoders and encoders have been executed using different deep learning architectures but since we are dealing with sequential data (one line per profile), the encoder in our model is uni-directional LSTM (Long Short Term Memory) whereas the decoder is AttentionalRNN (Recurrent Neural Network). LSTM was chosen because of its outstanding performance in the fields of speech recognition, language modeling, and translation.

III. Feedback Learning

In this paper, we define the feedback learning as a learning cycle in which the network is being trained by continuously receiving an input from the user and it will learn those patterns of correction. After training, it will start resolving those errors automatically without the external help from the user. To make it simpler, in the first feedback cycle, network after completing the first training will translate the results. Those results will be incorporated along with the training data in the second feedback cycle. With the objective of decreasing human involvement, we focus on achieving a system that auto-corrects the errors after the feedback cycle. So whenever the network encounters similar faults or misplaced information in the data sequence, it predicts the appropriate output by keeping track of the context.

IV. Methodology

For the implementation we have used the Tensorflow version of OpenNMT [7] which is an open source library.

A. Word Based Sequence-to-Sequence model

In this model, we use encoder-decoder architecture by treating post-IE data as a neural sequence translation problem. The model is based on a word level tokenization, the encoder considers each sentence as a sequence of words. The configuration of our network is explained as below:

1) Layers : 2 (uni-directional RNN LSTM encoder and Attentional RNN LSTM decoder)
2) Size of Layers : 512
3) Word Embedding Size : 512

The reason for using the two layered uni-directional RNN is because the private dataset was limited and small.

1) Preprocessing

We consider the dataset generated using Faker [3] as described in Section V-A1 and the first step is to build source and target vocabularies by specifying the size of the vocabulary. The data consists of parallel source and target data with one sentence per line and each of the fields are separated by spaces. Each line in the source file corresponds to the equivalent line in the target file. The source file consists of erroneous data from the information extraction and the target file consists of correct data which acts as ground truth. It indicates that the error data has to be translated into correct data.

2) Training

For the dataset we prepared, the configuration described above is used. The Values of the hyperparameters like learning rate, dropout percentage, batch size have been chosen after running many iterations and considering the problem of over-fitting in mind.

1) Dropout Percentage : 30
2) Optimizer : Adam Optimizer
3) Learning Rate : 0.001
4) Beam Width : 4
5) Batch Size : 32

B. Dictionary look-up using Hunspell

In this method, we tokenize the test dataset and pass each of the words to our custom dictionary using Hunspell [14] to correct the mistakes and select the best prediction. Here we use our own dictionary because the dataset in our experiment contains proper nouns and using a generic German dictionary would produce a bad result by default. Since our dataset has numerical values such as insurance number and date of birth, we correct this information using regular expression because any dictionary would not be able to predict the numerical mistakes.

V. Experimental Design

In this Section, we present the different kinds of datasets used for training and testing our network model. We discuss how each of the datasets is generated and give details of the test cases involved in the translation of the output. After the prediction, we evaluate the resulting output of our different test cases and analyze the performance of the model later. Figure 4 describes the schematic overview of Feedback Learning III cycle that has been incorporated for error correction of our dataset.

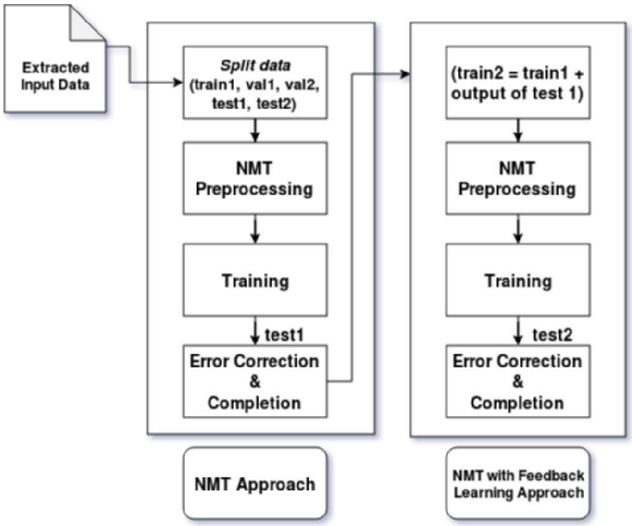

Extracted Input Data
Split data (train1, val1, val2, test1, test2)
NMT Preprocessing
Training
test1
Error Correction & Completion
(train2 = train1 + output of test 1)
NMT Preprocessing
Training
test2
Error Correction & Completion
NMT Approach
NMT with Feedback Learning Approach

Figure 4: The architectural representation of our Feedback Learning pipeline for error correction of the extracted information.

A. Datasets

1) Synthetic Use Case

This dataset is generated from an open source Python library named Faker [3]. We considered private dataset as a base and tried to replicate a similar format of data by generating data from the library. For this experiment, we created synthetic data having 150,000 user profiles with the corresponding fields *(FirstName, LastName, Address, Hospital Name, Hospital Address, Sex, Date of Birth, Phone Number, Insurance Number)*. In this data, there are 25,000 unique user profiles and the rest 125,000 are the replication from the unique profiles with different combinations of data fields mentioned above.

Now that we have ground truth, the challenge was to simulate erroneous data almost identical to OCR output. For this purpose, we used a document analysis tool Tesseract-OCR [15] to identify the statistics of common information extraction errors in a real-world scenario. From the character distribution stats through Tesseract, we generated error data by replacing certain characters from the ground truth with the identified errors. For example ('a' : 'o', 'e':'€'). We created artificial noise by corrupting 95% of the data generated using Faker with a variety of character replacements. We have assigned percentages to each of the particular characters which need to be replaced. By the end of this process, there will be two datasets, one with ground truth and the other with error data. For a real-world scenario, the erroneous data represents the extracted information coming out from a digital mailroom system, and the ground truth data represents the manually corrected information with the help of human verification. The sample information sequence, ground truth, and the respective error profile are mentioned below. It is important to note that the delimiter between the two consecutive information is space, however, it is also possible that space could occur within the information element, for example, the address might have multiple information such as street name, postcode, city, and country (Leonid-Renner-Platz 71165 Wurzen Hamburg Germany) in the sample.

Information sequence in our dataset follow this order **<Address, Birthdate, Blood Group, First Name, Insurance Number, Hospital Address, Hospital ID, Hospital Name, Hospital City, Hospital Postcode, Last Name, Phone Number, Gender>**

Ground Truth : *Leonid-Renner-Platz 71165 Wurzen Hamburg Germany 2004-06-08 A+ Reimar 422893598198 Sankt Annen Str.9 990702828 Krankenhaus St. Anna-Stift Löningen 49624 Wilms 03549 58413 M*

Error Data: *L€onjd-R€nn€r-Platz 71165 Wurz€n Hamburg, G€rmony 2004-06-0O A+ Reimar 0422893598198 $ankt Amen Str.9 960702328 Krankenhau$ St. Anna-Stift Löningen 49624 Wilm$ 03549 58413 M*

The dataset is divided into train, test and validation sets having two parallel documents (Ground truth and error data) since OpenNMT requires a source(error data) and target(ground truth) as an input during training. The distribution of the data into Training, Validation, and Testing is elaborated in the Table I.

2) Private Use Case

This dataset is rather a small one as compared to synthetic but it is based on the actual information from an insurance company which makes this a critical use case. Since the number of given unique profiles is only 94 which are certainly not enough to train a deep neural network so we increased the number of profiles by augmenting the data. Here, the input data is processed output of an OCR system and no artificial errors were introduced in this use case. This approach leads to 20,000 profiles where each profile is treated in a single sentence. The distribution of the data into Training, Validation, and Testing for this case is also explained in Table I.

Dataset Statistics for both of the use cases				
Datasets	# Sentences	# Training	# Validation	# Test
Synthetic	150,000	70,000	10,000	30,000
Private	20,000	12,000	1,000	3,000

Table I: Data distribution showing the number of samples used in each case for the first feedback learning cycle where one sample is a single user profile.

Table II illustrates the data distribution of feedback cycle 2 which is the second iteration. In this loop, we combine the prediction results obtained from the feedback cycle 1 with our training data. Hence, the size of the training set for synthetic use will be 100,000 samples (70,000 + 30,000 from feedback cycle 1). After updating the vocabularies in the preprocessing stage, we train the model again for further 10,000 steps in the synthetic use case and 5,000 more steps in private use case. While deciding the number of steps in the second feedback learning cycle, the evaluation loss is considered as a significant factor. Once the model is trained, we infer the test set from Table II on both of our models and calculate the prediction accuracy.

Dataset Statistics for both of the use cases				
Datasets	# Sentences	# Training	# Validation	# Test
Synthetic	150,000	100,000	10,000	30,000
Private	20,000	15,000	1,000	3,000

Table II: Data distribution showing the number of samples used in each case for the second feedback learning cycle where one sample is a single user profile.

VI. Performance Evaluation

In this Section, we present the results of the various test cases. This helps us to understand whether the predicted output of our deep learning trained network has improved or not. The following test cases were used :

- **Test Case 1**: In this experiment, we used the test set that was obtained by splitting the erroneous data.
- **Test Case 2**: We achieved this test data by introducing a new set of errors to Test Case 1. For example, now *s* can be replaced with *5* or *m* can be substituted with *rn* instead of *w*. We did this to check if the model still predicts the output sequence correctly even when it has not seen or trained on new kind of errors. ***Example :*** *Lconid-Rcnncr-Platz 77165 Wurzen Hamburg Gcrmany 2oo4-06-08 A+ Reiiimar 0422893598198 Sankt Annen Str.9 990702828 Krankenhaus St.Anna-Stift Leaningen 49624 Wiiilms 03549 58413 M*
- **Test Case 3**: In this experiment, we removed a few data fields from the ground truth that was generated initially.

Example: *Leonid-Renner-Platz 71165 Wurzen Hamburg Germany 2004-06-08 A+ Sankt Annen Str.9 990702828 Krankenhaus St. Anna-Stift Löningen 49624 Wilms 03549 58413 M*

- **Test Case 4** : In this experiment, we removed one random field on even line and two random fields if it's an odd line in the dataset.

  Example: *L€onjd-R€nn€r-Platz 71165 Hamburg G€rmany 2004-06-0O A+ Reimar 0422893598198 $ankt Amen Str.9 Krankenhau$ St.Anna-Stift Löningen 49624 Wilm$ 03549 58413 M*
- **Hunspell Case 5**: In this experiment, we have not used the default Hunspell dictionary [14] but used the customized version of Hunspell which has been explained in the methodology section (IV-B).

A. Evaluation Metrics

For an error correction and completion system, accuracy is defined as how identical is the predicted document with respect to the target document. We measured the quality of output using Levenshtein distance[4]. It helps in identifying the number of edit operations required to transform the error data to ground truth and it gives the similarity measure between two documents. In all of our test cases, we have used word-level error measurement because as described in Test Cases 3 & 4, the dataset had many missing information fields. We calculate the total number of operations using the formula:

$$WER = \frac{I + D + S}{N} \times 100 \tag{1}$$

In the formula 1, **N** represents the total number of words in the dataset, and **I** represents the minimum number of word insertion operation that was required to transform error sequence to the ground truth. Similarly, **D** represents word deletion and **S** represents word substitution.

B. Synthetic Use Case

Table III explains the results for the individual test cases which were elaborated earlier and it can be seen that NMT with feedback learning technique has outperformed the state of the art NMT method in all of the test cases. We have also tried the error correction using Hunspell on Test Case 1 but the results were poor because dataset consists of many proper nouns and we could only achieve **31.39%** accuracy. In Table III, the *Accuracy Before* represents the word level accuracy of the document with respect to the ground truth after adding artificial noise.

Figure [5] displays the evaluation loss of our training model. The training has been stopped after 15,000 steps since the evaluation loss starts ascending and could cause the model to overfit. The orange line indicates the training loss while the blue line implies the evaluation loss.

Accuracy of the Synthetic Test Cases			
Test Cases	Accuracy Before	NMT	Feedback NMT
1	27.14	93.21	**94.91**
2	27.03	92.47	**93.22**
3	38.80	91.48	**93.18**
4	21.08	90.89	**92.63**

Table III: Accuracy in each test case before and after applying the trained sequence to sequence model on extracted information.

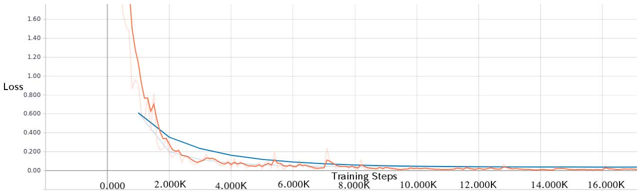


Figure 5: Training progress of the network after every 2,000 steps on Synthetic use case to monitor the evaluation loss. The training is stopped after 15,000 steps.

C. Private Use Case

For the private use case, we have also achieved good results. Table IV describes the result on post-OCR information extraction. In Table IV, the *Accuracy Before* is the word-level accuracy of the data directly taken from the OCR output. Even here, we can notice a good amount of increase in the final accuracy after the first feedback learning cycle.

Accuracy of the Private Test Case			
Test Cases	Accuracy Before	NMT	Feedback NMT
1	59.45	91.15	**92.61**

Table IV: Accuracy of the private use case before and after applying the trained sequence to sequence model on extracted information.

Figure [6] explains the evaluation loss of private use case and the training is done only until 10,000 steps as the dataset is small and it has least validation loss at that point.

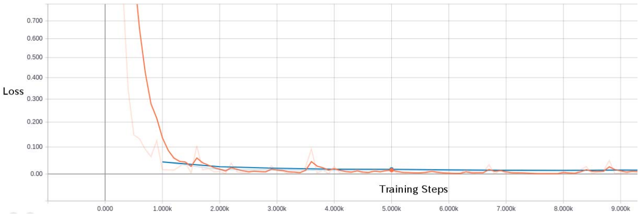


Figure 6: Training progress of the network after every 1,000 steps on Private use case to monitor the evaluation loss. The training is stopped after 9,000 steps

D. Results Comparison

In this Section, we compare the results of the Feedback Learning with NMT approach and the other pre-existing state of the art post-OCR error correction methodologies. In Table V, SMT is Statistical Machine Translation [1] whereas SA is Simulated Annealing [6]. It can be seen that our NMT with Feedback Learning approach has performed better than SMT and simulated annealing (SA) methods.

SMT	NMT	SA	NMT with Feedback
81.41	90.89	79.23	**92.63**

Table V: Result comparison of state-of-the-art approaches with our Feedback Learning method specifically for Test Case 4 where we have missing information fields.

E. Significance Test

We wanted to prove that the new model trained in the second feedback cycle having the prediction results from the first feedback cycle would give us better results as compared to the model trained in the first feedback cycle. In order to reject or accept the null hypothesis, we performed the significance test by applying reinforcement sampling technique and divided our synthetic test set of 30,000 samples into 10 pieces of 3,000 samples each and a private test set of 3,000 samples into 10 batches of 300 samples. We calculated the prediction accuracy by inferring each of these samples and our t-test value identifies that the new model produced after the second feedback cycle has a significant improvement in performance and predicted better results. This also proves our assumption and rejects the null hypothesis. It also helps us recognizing another point that retraining with the predicted output will always improve the results each time accounting to the feedback patterns and reduces the human involvement in correcting the errors with the increase in the size of the dataset.

VII. Conclusion

Post-IE error corrections have become a vital step for processing information from the graphical documents. In this paper, we proposed the new feedback learning technique that explains how erroneous words after information extraction can be corrected by reducing the human effort in the detection and correction of post-IE errors. We have implemented this concept through deep learning architecture using OpenNMT which is an open source tool. Our method manages to convert 27.14% accuracy of information extraction in test case 1 into 93.21% accuracy of information extraction after first feedback learning cycle and this accuracy is further increased up to 94.91% in the second feedback learning cycle in the synthetic use case. While in the private use case it converts 59.45% accuracy of information extraction for the test case 1 into 91.15% accuracy after first feedback learning cycle which is further improved to 92.61% in the second feedback learning cycle.

Our current results have involved word based tokenization; however, it would be interesting to explore the current approach using character-based tokenization. Taking the limited size of the dataset into consideration, we have used uni-directional LSTM. In case of a relatively bigger dataset, bi-directional LSTM can be used which may lead to even better performances. Exploiting this feedback learning approach on other supervised classification problems could be a thought-provoking idea.

References

[1] Haithem Afli et al. “Using SMT for OCR error correction of historical texts”. In: (2016).

[2] Kyunghyun Cho et al. “Learning phrase representations using RNN encoder-decoder for statistical machine translation”. In: *arXiv preprint arXiv:1406.1078* (2014).

[3] *Faker is a Python package that generates fake data for you.* URL: https://github.com/joke2k/faker.

[4] Wael H Gomaa and Aly A Fahmy. “A survey of text similarity approaches”. In: *International Journal of Computer Applications* 68.13 (2013), pp. 13–18.

[5] Mark A Jones and Jason M Eisner. “A probabilistic parser and its applications”. In: 1992.

[6] Gitansh Khirbat. “OCR Post-Processing Text Correction using Simulated Annealing (OPTeCA)”. In: 2017.

[7] Guillaume Klein et al. “OpenNMT: Open-Source Toolkit for Neural Machine Translation”. In: 2017.

[8] Karen Kukich. “Techniques for automatically correcting words in text”. In: (1992).

[9] William B Lund, Douglas J Kennard, and Eric K Ringger. “Combining multiple thresholding binarization values to improve OCR output”. In: International Society for Optics and Photonics. 2013.

[10] William B Lund and Eric K Ringger. “Improving optical character recognition through efficient multiple system alignment”. In: ACM. 2009.

[11] William B Lund and Eric K Ringger. “Improving optical character recognition through efficient multiple system alignment”. In: ACM. 2009.

[12] William B Lund, Eric K Ringger, and Daniel D Walker. “How well does multiple OCR error correction generalize?” In: International Society for Optics and Photonics. 2014.

[13] Kareem Mokhtar, Syed Saqib Bukhari, and Andreas Dengel. “OCR Error Correction: State-of-the-Art vs an NMT-based Approach”. In: IEEE. 2018.

[14] Lszl Nmeth. *The most popular spellchecking library.* URL: https://github.com/hunspell/hunspell.

[15] Ray Smith. “An overview of the Tesseract OCR engine”. In: *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007).* Vol. 2. IEEE. 2007, pp. 629–633.

[16] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. “Sequence to sequence learning with neural networks”. In: 2014.

[17] Brian Tubay and Marta R Costa-jussà. “Neural machine translation with the transformer and multi-source romance languages for the biomedical WMT 2018 task”. In: 2018.